

[illegible]LN
LN
LNLN
LN
LN

LN
LN
LN
LN
LN
LN

LN

LN

LN
LNLN
LNLN
LNLN
LNLN
LNLN
LN

LN

LN

LN

LN
LNLN
LN

LN

100

100

```

LL               IIIIII               SSSSSSSS
LL               IIIIII               SSSSSSSS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SSSSSS
LL               II                   SSSSSS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LLLLLLLLLLLL    IIIIII               SSSSSSSS
LLLLLLLLLLLL    IIIIII               SSSSSSSS

```

LN
VO

.....


```
0001 0 module lnk_objpass1
0002 0      (ident = 'V04-000'
0003 0      ,addressing_mode
0004 0      (external = general
0005 0      ,nonexternal = long_relative
0006 0      ) =
0007 0
0008 1 begin
0009 1
0010 1
0011 1 *****
0012 1 *
0013 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0014 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0015 1 *  ALL RIGHTS RESERVED.
0016 1 *
0017 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0018 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0019 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0020 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0021 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0022 1 *  TRANSFERRED.
0023 1 *
0024 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0025 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0026 1 *  CORPORATION.
0027 1 *
0028 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0029 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0030 1 *
0031 1 *
0032 1 *****
0033 1
0034 1 ++
0035 1
0036 1 MODULE:      LNK_OBJPASS1
0037 1
0038 1 FACILITY:    LINKER
0039 1
0040 1 ABSTRACT:    Pass one of all object modules of this link.
0041 1
0042 1 HISTORY:
0043 1
0044 1     VERSION: X01.00
0045 1
0046 1     AUTHOR: T.J. PORTER 30-DEC-76
0047 1
0048 1 MODIFIED BY:
0049 1
0050 1     V04-045 JWT0182      Jim Teague      14-May-1984
0051 1     Adjust logic in PROSYMBOL so that lnk$gw_nudfsyms
0052 1     doesn't get decremented too many times.
0053 1
0054 1     V04-044 ADE0003      Alan D. Eldridge  4-May-1984
0055 1     Fix access violation in routine PROSYMBOL.
0056 1
0057 1     V04-043 ADE0002      Alan D. Eldridge  1-May-1984
```

58	0058	1	Fix bug in reporting 'illegal module name length'.
59	0059	1	
60	0060	1	V04-042 ADE0001 Alan D. Eldridge 3-Mar-1984
61	0061	1	If GSMATCH high bit is set, then ignore date comparison
62	0062	1	between shareable image library entry and shareable image
63	0063	1	itself.
64	0064	1	
65	0065	1	V04-041 JWT0154 Jim Teague 9-Feb-1984
66	0066	1	MPC blocks are not always clean when memory is
67	0067	1	allocated for them. This caused ridiculous values
68	0068	1	for psect length to show up in the data produced
69	0069	1	in the Debugger Module/Psect Table (DMT).
70	0070	1	
71	0071	1	V03-040 JWT0132 Jim Teague 16-Aug-1983
72	0072	1	Adjust ordering of LNKOPT fdb's that are created.
73	0073	1	A module that specifies several files in LNKOPT
74	0074	1	records caused the linker to append the LNKOPT
75	0075	1	file fdb's in reverse order!
76	0076	1	
77	0077	1	V03-039 JWT0130 Jim Teague 28-Jul-1983
78	0078	1	More scrutiny for Linker Options Records.
79	0079	1	
80	0080	1	V03-038 JWT0128 Jim Teague 24-Jul-1983
81	0081	1	Fix environment ref/def problem.
82	0082	1	
83	0083	1	V03-037 JWT0125 Jim Teague 23-Jun-1983
84	0084	1	Reincarnate univ=* when accompanied by /noexe.
85	0085	1	
86	0086	1	V03-036 JWT0121 Jim Teague 20-May-1983
87	0087	1	Sound the death knell for UNIVERSAL=*
88	0088	1	
89	0089	1	V03-035 JWT0113 Jim Teague 20-Apr-1983
90	0090	1	Filter universal bit out of incoming symbols;
91	0091	1	Call \$getjpi to get number of open files left.
92	0092	1	
93	0093	1	V03-034 JWT0109 Jim Teague 13-Apr-1983
94	0094	1	Adjust code for Linker Options Record.
95	0095	1	
96	0096	1	V03-033 JWT0099 Jim Teague 14-Mar-1983
97	0097	1	New CLI interface. Also implemented code for
98	0098	1	Linker Options Record.
99	0099	1	
100	0100	1	V03-032 JWT0071 Jim Teague 02-Dec-1982
101	0101	1	Added NAME and IDENTIFICATION options.
102	0102	1	
103	0103	1	V03-031 JWT0061 Jim Teague 22-Oct-1982
104	0104	1	Add debugger image section to images linked /debug.
105	0105	1	
106	0106	1	V03-030 JWT0052 Jim Teague 15-Sep-1982
107	0107	1	Fix bug which caused option-defined symbols to be
108	0108	1	excluded from STB.
109	0109	1	
110	0110	1	V03-029 JWT0047 Jim Teague 06-Aug-1982
111	0111	1	Fix the linker psect alignment bug, once and for all!
112	0112	1	
113	0113	1	V03-028 JWT0045 Jim Teague 30-Jul-1982
114	0114	1	Fix bug in entity id mismatch error message.


```

: 115 0115 1 :
: 116 0116 1 :
: 117 0117 1 :
: 118 0118 1 :
: 119 0119 1 :
: 120 0120 1 :
: 121 0121 1 :
: 122 0122 1 :
: 123 0123 1 :
: 124 0124 1 :
: 125 0125 1 :
: 126 0126 1 :
: 127 0127 1 :
: 128 0128 1 :
: 129 0129 1 :
: 130 0130 1 :
: 131 0131 1 :
: 132 0132 1 :--

```

```

V03-027 JWT0043      Jim Teague      15-Jul-1982
      Fix psect alignment bug which caused all contributions to
      a psect to have maximized alignment.

V03-026 JWT0039      Jim Teague      23-Jun-1982
      Add mask in order to retain SYMSV_LCLSYM attribute.
      This keeps module local symbols out of cross references.

V03-025 JWT0033      Jim Teague      25-May-1982
      Or OMD$V_NOBINS for each FDB and propagate to FDB$V_OMDNOBIN
      Use option-specified psect alignment if present in PROPSECTDEF

V03-024 BLS0161      Benn Schreiber   19-Mar-1982
      Correct V03-021 handling of selective search procedures

```



```
134 0133 1 ++
135 0134 1
136 0135 1 FUNCTIONAL DESCRIPTION:
137 0136 1
138 0137 1 THIS ROUTINE IS CALLED TO PROCESS ALL OBJECT MODULES
139 0138 1 DURING PASS ONE OF THE LINK. IT VALIDATES OBJECT MODULE
140 0139 1 FORMAT, READS THE MODULE HEADER, ALL GSD AND THE
141 0140 1 END OF MODULE RECORDS AND BUILDS SYMBOL AND P-SECTION
142 0141 1 TABLES. IT IS IN THIS ROUTINE ALONE THAT THE LINKER
143 0142 1 DETERMINES WHETHER THE MODULE IS CONCATENATED. IT IS
144 0143 1 ALSO HERE THAT ANY USER DEFINED TRANSFER ADDRESS IS
145 0144 1 EXTRACTED.
146 0145 1
147 0146 1 THIS ROUTINE IGNORES THE CONTENT OF TIR, DBG AND LNK RECORDS.
148 0147 1
149 0148 1 --
150 0149 1 require 'PREFIX';
151 0264 1 forward routine
152 0265 1   prolnkrec,
153 0266 1   checkpddentry,
154 0267 1   lnk$insudfsym : novalue,
155 0268 1   seqchk,
156 0269 1   prohdr,
157 0270 1   delete_psect,
158 0271 1   findpsectmapent,
159 0272 1   alloc_omdnode,
160 0273 1   lnk$compare_omd,
161 0274 1   propsectdef,
162 0275 1   crefilter,
163 0276 1   prosymbol,
164 0277 1   symbols,
165 0278 1   entpnts,
166 0279 1   procedef,
167 0280 1   proeom,
168 0281 1   countdbg,
169 0282 1   compare_idc,
170 0283 1   alloc_idc,
171 0284 1   randentity,
172 0285 1   compare_env,
173 0286 1   alloc_env,
174 0287 1   insudfenv : novalue,
175 0288 1   lnk$findenvmap,
176 0289 1   proenv,
177 0290 1   localsymbols,
178 0291 1   localentpnt,
179 0292 1   localprocedef,
180 0293 1   progsd,
181 0294 1   lnk$procsobj,
182 0295 1   lnk$objpass1 ;
183 0296 1
184 0297 1 library 'LIBL32' ;
185 0298 1
186 0299 1 library 'DATBAS' ;
187 0300 1
188 0301 1 require 'ISGENC' ;
189 0685 1 !
190 0686 1 !
```

PROCESS LINKER OPTIONS RECORD
CHECK PSECT DEF. ENTRY AND DEFINE IT
INSERT SYMBOL IN UNDEFINED SYMBOL LIST
CHECK RECORD SEQUENCE
PROCESS MODULE HEADER RECORDS
DELETE SHAREABLE IMAGE PSECT FROM OTHER CLUSTERS
FIND PSECT MAPPING TABLE ENTRY
ALLOCATE OMD NODE BLOCK
COMPARE OMD NODE BLOCKS
PROCESS PSECT DEFINITIONS
CROSS REFERENCE FILTER FOR SYMBOLS
PROCESS SYMBOLS
FRONT END FOR SYMBOL DEFS/REFS
FRONT END FOR ENTRY POINTS
FRONT END FOR PROCEDURE DEFINITIONS
PROCESS END OF MODULE RECORDS
COUNT DEBUG AND TRACEBACK RECORDS
COMPARE RANDOM IDENT BLOCKS
ALLOCATE RANDOM IDENT BLOCK
PROCESS RANDOM IDENT
COMPARE ENVIRONMENT BLOCKS
ALLOCATE ENVIRONMENT BLOCK
INSERT UNDEFINED ENVIRONMENT
FIND ENVIRONMENT MAPPING TABLE ENTRY
PROCESS ENVIRONMENT
FRONT END FOR LOCAL SYMBOLS
FRONT END FOR LOCAL ENTRY POINTS
FRONT END FOR LOCAL PROCEDURE DEFS
PROCESS GSD RECORDS
PROCESS AN OBJECT MODULE
DO PASS 1

! SYSTEM DATA STRUCTURES
! INTERNAL DATA BASE DEFINITIONS
! IMAGE SECTION DEFINITIONS AND OBJECT LANG. DEFS.

```
: 191      0687 1 ! ENSURE THAT SYMS$ SYMBOLS CORRESPOND TO GSY$ SYMBOLS
: 192      0688 1 ! AND THE PSC$ SYMBOLS CORRESPOND TO GPSS$ SYMBOLS
: 193      0689 1 !
: 194      0690 1 %if (sym$m_weak and gsy$m_weak) eql 0 %then %warn ('SYMS$ WEAK NEQ GSY$ WEAK') %fi;
: 195      0691 1 %if (sym$m_def and gsy$m_def) eql 0 %then %warn ('SYMS$ DEF NEQ GSY$ DEF') %fi;
: 196      0692 1 %if (sym$m_uni and gsy$m_uni) eql 0 %then %warn ('SYMS$ UNI NEQ GSY$ UNI') %fi;
: 197      0693 1 %if (sym$m_rel and gsy$m_rel) eql 0 %then %warn ('SYMS$ REL NEQ GSY$ REL') %fi;
: 198      0694 1 %if (psc$m_pic and gps$m_pic) eql 0 %then %warn ('PSC$ PIC NEQ GPSS$ PIC') %fi;
: 199      0695 1 %if (psc$m_lib and gps$m_lib) eql 0 %then %warn ('PSC$ LIB NEQ GPSS$ LIB') %fi;
: 200      0696 1 %if (psc$m_ovr and gps$m_ovr) eql 0 %then %warn ('PSC$ OVR NEQ GPSS$ OVR') %fi;
: 201      0697 1 %if (psc$m_rel and gps$m_rel) eql 0 %then %warn ('PSC$ REL NEQ GPSS$ REL') %fi;
: 202      0698 1 %if (psc$m_gbl and gps$m_gbl) eql 0 %then %warn ('PSC$ GBL NEQ GPSS$ GBL') %fi;
: 203      0699 1 %if (psc$m_shr and gps$m_shr) eql 0 %then %warn ('PSC$ SHR NEQ GPSS$ SHR') %fi;
: 204      0700 1 %if (psc$m_exe and gps$m_exe) eql 0 %then %warn ('PSC$ EXE NEQ GPSS$ EXE') %fi;
: 205      0701 1 %if (psc$m_rd and gps$m_rd) eql 0 %then %warn ('PSC$ RD NEQ GPSS$ RD') %fi;
: 206      0702 1 %if (psc$m_wrt and gps$m_wrt) eql 0 %then %warn ('PSC$ WRT NEQ GPSS$ WRT') %fi;
: 207      0703 1 %if (psc$m_vec and gps$m_vec) eql 0 %then %warn ('PSC$ VEC NEQ GPSS$ VEC') %fi;
: 208      0704 1
: 209      0705 1 global literal
: 210      0706 1     psc$m_pscbits =      psc$m_pic or psc$m_lib or psc$m_ovr      ! PSECT ATTRIBUTES ALL TOGETHER
: 211      0707 1                      or psc$m_rel or psc$m_gbl or psc$m_shr
: 212      0708 1                      or psc$m_exe or psc$m_rd or psc$m_wrt
: 213      0709 1                      or psc$m_vec,
: 214      0710 1
: 215      0711 1     psc$m_shrbits =      psc$m_gbl or psc$m_shr or psc$m_ovr ; ! PSECTS CONSIDERED FOR SHAREABLE IMAGE
```



```
: 217      0712 1 external routine
: 218      0713 1   lnk$closefile,
: 219      0714 1   lnk$filnamdsc,
: 220      0715 1   lnk$openlib,
: 221      0716 1   lnk$allofdb,
: 222      0717 1   crf$insrtkey,
: 223      0718 1   crf$insrtref,
: 224      0719 1   lib$insert_tree,
: 225      0720 1   lib$lookup_tree,
: 226      0721 1   lib$straverse_tree,
: 227      0722 1   lnk$compare_pscnod,
: 228      0723 1   lnk$srcpscdef,
: 229      0724 1   lnk$bintim,
: 230      0725 1   lnk$alloblk      : novalue,
: 231      0726 1   lnk$dealblk      : novalue,
: 232      0727 1   lnk$exit         : novalue,
: 233      0728 1   lnk$findpscnam,
: 234      0729 1   lnk$insert,
: 235      0730 1   lnk$nextfil,
: 236      0731 1   lnk$nextrec,
: 237      0732 1   lnk$procshrim,
: 238      0733 1   lnk$proclib,
: 239      0734 1   lnk$search,
: 240      0735 1   lnk$searchlocal,
: 241      0736 1   lnk$addimage;
: 242      0737 1
: 243      0738 1 external literal
: 244      0739 1   lib$_keyalrins,
: 245      0740 1   lin$_openin,
: 246      0741 1   lin$_badccc,
: 247      0742 1   lin$_badpsc,
: 248      0743 1   lin$_datmismch,
: 249      0744 1   lin$_eomftl,
: 250      0745 1   lin$_errors,
: 251      0746 1   lin$_excpsc,
: 252      0747 1   lin$_fatalerror,
: 253      0748 1   lin$_format,
: 254      0749 1   lin$_gsdtyp,
: 255      0750 1   lin$_entidmtch,
: 256      0751 1   lin$_entidmtchb,
: 257      0752 1   lin$_entidmtcho,
: 258      0753 1   lin$_entidmtcht,
: 259      0754 1   lin$_illfmlcnt,
: 260      0755 1   lin$_illnamelen,
: 261      0756 1   lin$_illrectyp,
: 262      0757 1   lin$_illreclen,
: 263      0758 1   lin$_modnam,
: 264      0759 1   lin$_muldef,
: 265      0760 1   lin$_muldefpsc,
: 266      0761 1   lin$_mulpssc,
: 267      0762 1   lin$_multfr,
: 268      0763 1   lin$_mulshrpssc,
: 269      0764 1   lin$_noeom,
: 270      0765 1   lin$_noimgfil,
: 271      0766 1   lin$_nomods,
: 272      0767 1   lin$_nopscs,
: 273      0768 1   lin$_nudfenvs,
```

```
! INSERT KEY IN CREF TABLE
! INSERT REF TO KEY IN CREF TABLE
! INSERT INTO BINARY TREE
! LOOKUP IN BINARY TREE
! TRAVERSE BINARY TREE
! COMPARE PSECT NAME WITH GPSLST NODE
! FIND PSECT DEFINED IN OPTION FILE
! CONVERT DATE/TIME TO BINARY
! DYNAMIC MEMORY ALLOCATION
! AND DEALLOCATION
! EXIT ROUTINE
! SEARCH FOR A P-SECTION ENTRY BY NAME
! SYMBOL TABLE INSERTION
! GET NEXT INPUT FILE
! OBTAIN NEXT RECORD OF OBJECT MODULE
! PROCESS SHAREABLE IMAGES
! PROCESS AN OBJECT LIBRARY FILE
! SYMBOL TABLE SEARCH
! MODULE-LOCAL SYMBOL TABLE SEARCH
! ADD SHAREABLE IMAGE CLUSTER
```

```
! KEY ALREADY INSERTED IN TREE
```

```
! BAD COMPILER COMPLETION CODE
! ILLEGAL P-SECTION REFERENCED
! CREATION DATE/TIME MISMATCH
! EOM RECORD SPECIFIES ABORT
! COMPILATION HAD FATAL ERRORS
! TOO MANY PSECTS DEFINED
! FATAL ERROR MESSAGE ISSUED
! OBJECT MODULE FORMAT ERROR
! ILLEGAL GSD TYPE
! ASCIC IDENT MISMATCH
! BINARY IDENT MISMATCH
! IDENT OBJECT TYPE MISMATCH
! IDENT TYPE MISMATCH
! ILLEGAL FORMAL ARGUMENT COUNTS
! ILLEGAL NAME LENGTH
! ILLEGAL RECORD TYPE
! ILLEGAL RECORD LENGTH
! ILLEGAL MODULE NAME
! MULTIPLE SYMBOL DEFINITION
! MULT. DEF OF PSECT IN SAME MODULE
! MULTIPLE PSECT ATTRIBUTES DEF.
! MULTIPLE TRANSFER ADDRESSES
! MULTIPLY DEFINED SHAREABLE IMAGE PSECT
! MISSING END OF MODULE ERROR
! IMAGE FILE NOT CREATED
! NO MODULES FOUND (IN THE LIBRARIES)
! NO PROGRAM SECTIONS FOUND
! NUMBER OF UNDEFINED ENVIRONMENTS
```


274	0769	1	lin\$_nudfsyms,	!	NUMBER OF UNDEFINED SYMBOLS
275	0770	1	lin\$_nudflsyms,	!	NUMBER OF UNDEFINED LOCAL SYMBOLS
276	0771	1	lin\$_ovrali,	!	OVERLAYED P-SECTION WITH DIFFERENT ALIGNMENT
277	0772	1	lin\$_pscali,	!	ILLEGAL P-SECTION ALIGNMENT
278	0773	1	lin\$_pscnxr,	!	P-SECTION CONTAINING TRANSFER ADDRESS IS NOT EXE/REL
279	0774	1	lin\$_reclng,	!	ILLEGAL RECORD LENGTH
280	0775	1	lin\$_rectyp,	!	ILLEGAL RECORD TYPE
281	0776	1	lin\$_seqnce,	!	RECORDS IN ILLEGAL SEQUENCE
282	0777	1	lin\$_shrp\$clng,	!	PSECT IN OBJ MOD BIGGER THAN ONE IN SHR IMAGE
283	0778	1	lin\$_strlvl,	!	ILLEGAL STRUCTURE LEVEL IN MODULE
284	0779	1	lin\$_udfenv,	!	LIST AN UNDEFINED ENVIRONMENT
285	0780	1	lin\$_udfsym,	!	EACH UNDEFINED SYMBOL
286	0781	1	lin\$_wners,	!	COMPILER ISSUED WARNINGS
287	0782	1	lnk\$_k_max_filename_length ;	!	MAXIMUM FILENAME LENGTH
288	0783	1			
289	0784	1	external		
290	0785	1	lnk\$_gt_ipilst,		
291	0786	1	lnk\$_gl_filesleft,		
292	0787	1	lnk\$_al_rab	: block [rab\$_c_bln, byte],	! RAB USED TO OPEN THE FILE
293	0788	1	lnk\$_al_sytblfmt,		! SYMBOL LISTING FORMAT DESCRIPTION TABLE ADDRESS
294	0789	1	lnk\$_al_valctlb,		! CREF BY VALUE CONTROL TABLE ADDRESS
295	0790	1	lnk\$_aw_version	: block [lid\$_c_size, byte],	! VERSION ARRAY
296	0791	1	lnk\$_gl_ctlmsk	: block [, byte],	! LINK CONTROL MASK
297	0792	1	lnk\$_gl_pshnum,		! NUMBER OF PSECT TO BE OUTPUT TO SHAREABLE IMAGE
298	0793	1	lnk\$_gt_imgsta	: block [, byte],	! NAME OF "SYSSIMGSTA"
299	0794	1	lnk\$_gl_clulst	: vector [2],	! CLUSTER DESCRIPTOR LISTHEAD
300	0795	1	lnk\$_gl_defclu	: block [, byte],	
301	0796	1	lnk\$_gl_record,		! RECORD NUMBER IN THE FILE
302	0797	1	lnk\$_gl_shrsyms,		! NUMBER OF SHAREABLE IMAGE SYMBOLS REFERENCED
303	0798	1	lnk\$_gl_shrimgs,		! NUMBER OF SHAREABLE IMAGES REFERENCED
304	0799	1	lnk\$_gl_pscdflst,		! PSECTS DEFINED BY OPTION LIST
305	0800	1	lnk\$_gl_maxsymsz,		! MAXIMUM SYMBOL LENGTH SEEN
306	0801	1	lnk\$_gl_maxmodsz,		! MAXIMUM MODULE LENGTH SEEN
307	0802	1	lnk\$_gl_curclu	: ref block [, byte],	! CURRENT CLUSTER POINTER
308	0803	1	lnk\$_gl_curfil	: ref block [, byte],	! CURRENT FDB POINTER
309	0804	1	lnk\$_gl_curomd	: ref block [, byte],	! CURRENT OMD POINTER
310	0805	1	lnk\$_gl_libsym	: ref block [, byte],	! POINTS TO SYMBOL CAUSING LOAD OF LIBRARY MODULE
311	0806	1	lnk\$_gt_imgid	: vector [, byte],	! STORAGE OF IMAGE IDENT
312	0807	1	shrfiletype,		! .EXE string
313	0808	1	defiletype,		! .OBJ string
314	0809	1	libfiletype,		! .OLB string
315	0810	1			
316	0811	1	global		
317	0812	1	lnk\$_gl_cmddst	: initial (0)	! COUNT OF OBJMODS WHICH CONTRIBUTE TO DST
318	0813	1	lnk\$_gw_pscdst	: word initial (0),	! COUNT OF PSECTS IN THE ABOVE OBJMODS
319	0814	1	lnk\$_gl_udflst	: vector [2] initial (	! UNDEFINED SYMBOL LISTHEAD
320	0815	1		lnk\$_gl_udflst,	! INITIALLY POINTS TO
321	0816	1		lnk\$_gl_udflst),	! ITSELF, SINCE EMPTY
322	0817	1	lnk\$_gl_envtree,		! TREE HEAD FOR ENVIRONMENTS
323	0818	1	lnk\$_gl_udfenv	: vector [2] initial (	! UNDEFINED ENVIRONMENT LISTHEAD
324	0819	1		lnk\$_gl_udfenv,	
325	0820	1		lnk\$_gl_udfenv),	
326	0821	1	lnk\$_gl_udflsy	: vector [2] initial (	! LIST HEAD FOR UNDEFINED LOCAL SYMBOLS
327	0822	1		lnk\$_gl_udflsy,	
328	0823	1		lnk\$_gl_udflsy),	
329	0824	1	lnk\$_gl_entitree,		! TREE HEAD FOR ENTITIES IDENT CHECK
330	0825	1	lnk\$_gl_dbgtr,		! DEBUGGER TRANSFER ADDRESS


```
331 0826 1 lnk$gl_dbgtfps : ref block [, byte], : POINTER TO PSECT CONTAINING IT
332 0827 1 lnk$gl_tfradr, : TRANSFER ADDRESS
333 0828 1 lnk$gl_tfrpsc : ref block [, byte], : POINTER TO PSECT CONTAINING IT
334 0829 1 lnk$gl_dbgestim, : ESTIMATE OF NUMBER OF BYTES FOR DST
335 0830 1 lnk$gl_omdtree, : TREE HEAD FOR OBJ MODULE DESCRIPTORS
336 0831 1 lnk$gw_dbgrecs : word, : NUMBER OF DEBUG RECORDS
337 0832 1 lnk$gw_nfiles : word, : NUMBER OF FILES OPENED
338 0833 1 lnk$gw_nmodules : word, : NUMBER OF MODULES FOUND
339 0834 1 lnk$gw_npsects : word, : NUMBER OF PSECTS IN TABLE
340 0835 1 lnk$gw_nsymbols : word, : NUMBER OF SYMBOLS IN TABLE
341 0836 1 lnk$gw_ksymbols : word, : NUMBER OF SYMBOLS IN THIS IMAGE
342 0837 1 lnk$gw_ncrossrfs : word, : NUMBER OF CROSS REFERENCES ENTERED
343 0838 1 lnk$gw_nudfsyms : word, : NUMBER OF UNDEFINED SYMBOLS
344 0839 1 lnk$gw_nlsyms : word, : NUMBER OF MODULE-LOCAL SYMBOLS
345 0840 1 lnk$gw_nudflsyms : word, : NUMBER OF UNDEFINED MODULE-LOCAL SYMBOLS
346 0841 1 lnk$gw_nudfenvs : word ; : NUMBER OF UNDEFINED ENVIRONMENTS
347 0842 1
348 0843 1 own
349 0844 1 omd_has_dbg : byte initial (0), : CURRENT OBJMOD HAS DBG RECORDS
350 0845 1 lastobjmod : ref block [, byte], : POINTER TO LAST OBJ MODULE DESCRIPTOR
351 0846 1 weaktfradr : byte, : TRUE IF CURRENT XFR ADDR IS WEAK
352 0847 1 lclsymgsd : byte, : TRUE IF GSD SUBTYPE IS LOCAL SYMBOL
353 0848 1 wordpsectgsd : byte, : TRUE IF GSD SUBTYPE HAS WORD OF PSECT NUMBER
354 0849 1 mhdseen : byte initial (0), : FLAG MHD SUB-HEADER AS YET NOT SEEN
355 0850 1 lnmseen : byte initial (0), : FLAG THAT COMPILER NAME SUB-HEADER IS SEEN
356 0851 1 lastrectyp : byte, : TYPE OF THE PREVIOUS RECORD
357 0852 1 currectyp : byte initial (obj$c_eom), : TYPE OF THE CURRENT RECORD
358 0853 1 gsdooffset : word, : OFFSET INTO CONCATENATED GSD RECORD
359 0854 1 entrymask : word, : ENTRY POINT MASK OF CURRENT SYMCOL
360 0855 1 maxreclng : word initial (obj$c_maxreclsz), : ! MAXIMUM LENGTH PERMISSIBLE IN THIS MODULE
361 0856 1 objrecdesc : block [dsc$c_s_bln, byte], : STRING DESCRIPTOR FOR OBJECT RECORD
362 0857 1 symbolstring : ref vector [, byte], : POINTER TO CURRENT SYMBOL
363 0858 1 symtabent : ref block [, byte], : POINTER TO SYMBOL TABLE ENTRY
364 0859 1 symentnam : ref block [, byte], : POINTER TO SYMBOL NAME BLOCK
365 0860 1
366 0861 1 imageidstring : vector [sym$c_maxlng+1, byte], : DEFAULT IMAGE IDENT STORAGE
367 0862 1 obmodesc : block [omd$c_size, byte], : STATIC COPY OF OBJECT MODULE DESCRIPTOR
368 0863 1 last_lnkopt_fdb : ref block [, byte] ; : FDB OF LAST LNKOPT FILE FOR THIS MODULE
369 0864 1
370 0865 1 bind
371 0866 1 reclng = objrecdesc [dsc$w_length] : word, : SIZE OF RECORD READ
372 0867 1 objrec = objrecdesc [dsc$a_pointer] : ref block [, byte], : NAME POINTER PART OF DESCRIPTOR
373 0868 1 objvec = objrecdesc [dsc$a_pointer] : ref vector [, byte], : MUST ALSO ACCESS RECORD AS BYTE VECTOR
374 0869 1 recdispatch = plit ( : SET UP MAXIMUM ALLOWED RECORD TYPE (i.e.,
375 0870 1 : 0 - MODULE HEADER
376 0871 1 : 1 - GSD RECORDS
377 0872 1 : 2 - TIR - JUST CHECK CORRECT SEQUENCE
378 0873 1 : 3 - END OF MODULE
379 0874 1 : 4 - DBG - CHECK SEQUENCE AND COUNT RECORDS
380 0875 1 : 5 - TBT - COUNT AND CHECK SEQUENCE O.K.
381 0876 1 : 6 - OPTIONS RECORD
382 0877 1 :
383 0878 1 global bind
384 0879 1 lnk$gt_envstring = cstring ('Environment'),
385 0880 1 lnk$gt_modstring = cstring ('Module'), : FOR THE ERROR MESSAGE
386 0881 1 lnk$gt_symstring = cstring ('Symbol'),
387 0882 1 lnk$gt_pscstring = cstring ('Psect'),
```



```
: 388      0883 1      lnk$gt_clustring = cstring ('Cluster'),
: 389      0884 1      lnk$gt_entity    = cstring ('Entity'),
: 390      0885 1      lnk$gt_ident    = cstring ('Ident'),
: 391      0886 1      lnk$gt_objnam   = cstring ('Entity type'),
: 392      0887 1      lnk$gt_sysver   = cstring ('SYS$K_VERSION');
: 393      0888 1
: 394      0889 1 !
: 395      0890 1 routine dummy = return 1 ;
```

! SYSTEM VERSION SYMBOL
! FORCE BLISS TO PRINT THE

```
                                .TITLE LNK_OBJPASS1
                                .IDENT  \V04-000\
                                .PSECT  $SPLITS,NOWRT,NOEXE,2
                                .LONG   7
00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V 00000000V
                                .ADDRESS PROHDR, PROGSD, SEQCHK, PROEOM, -
                                COUNTDBG, COUNTDBG, PROLNKREC
                                .P.AAA: .BYTE 11
                                .P.AAB: .ASCII \Environment\
                                .P.AAC: .BYTE 6
                                .P.AAD: .ASCII \Module\
                                .P.AAE: .BYTE 6
                                .P.AAF: .ASCII \Symbol\
                                .P.AAG: .BYTE 5
                                .P.AAH: .ASCII \Psect\
                                .P.AAI: .BYTE 7
                                .P.AAJ: .ASCII \Cluster\
                                .P.AAK: .BYTE 6
                                .P.AAL: .ASCII \Entity\
                                .P.AAM: .BYTE 5
                                .P.AAN: .ASCII \Ident\
                                .P.AAO: .BYTE 11
                                .P.AAP: .ASCII \Entity type\
                                .P.AAQ: .BYTE 13
                                .P.AAR: .ASCII \SYS$K_VERSION\
                                .PSECT  $OWNS,NOEXE,2
                                00 00000 OMD_HAS_DBG:
                                .BYTE 0
                                00001 .BLKB 3
                                00004 LASTOBJMOD:
                                .BLKB 4
                                00008 WEAKTFRADR:
                                .BLKB 1
                                00009 LCLSYMGS:
                                .BLKB 1
                                0000A WORDPSECTGSD:
                                .BLKB 1
                                00 0000B MHDSEEN: .BYTE 0
                                00 0000C LNMSEEN: .BYTE 0
                                0000D LASTRECTYP:
                                .BLKB 1
                                03 0000E CURRECTYP:
                                .BYTE 3
                                0000F .BLKB 1
```



```
00010 GSDOFFSET:
      .BLKB 2
00012 ENTRYMASK:
      .BLKB 2
0800 00014 MAXRECLNG:
      .WORD 2048
00016      .BLKB 2
00018 OBJRECDESC:
      .BLKB 8
00020 SYMBOLSTRING:
      .BLKB 4
00024 SYMTABENT:
      .BLKB 4
00028 SYMENTNAM:
      .BLKB 4
0002C IMAGEIDSTRING:
      .BLKB 32
0004C OBMODESC:
      .BLKB 2120
00894 LAST_LNKOPT_FDB:
      .BLRB 4
      .PSECT $GLOBAL$,NOEXE,2
00000000 00000 LNK$GL_OMDDST::
      .LONG 0
0000 00004 LNK$GW_PSCDST::
      .WORD 0
00006      .BLKB 2
00000000' 00000000' 00008 LNK$GL_UDFLST::
      .ADDRESS LNK$GL_UDFLST, LNK$GL_UDFLST
00010 LNK$GL_ENVTREE::
      .BLKB 4
00000000' 00000000' 00014 LNK$GL_UDFENV::
      .ADDRESS LNK$GL_UDFENV, LNK$GL_UDFENV
00000000' 00000000' 0001C LNK$GL_UDFLSY::
      .ADDRESS LNK$GL_UDFLSY, LNK$GL_UDFLSY
00024 LNK$GL_ENTITREE::
      .BLKB 4
00028 LNK$GL_DBGTFR::
      .BLKB 4
0002C LNK$GL_DBGTFPS::
      .BLKB 4
00030 LNK$GL_TFRADR::
      .BLKB 4
00034 LNK$GL_TFRPSC::
      .BLKB 4
00038 LNK$GL_DBGESTIM::
      .BLKB 4
0003C LNK$GL_OMDTREE::
      .BLKB 4
00040 LNK$GW_DBGRECS::
      .BLKB 2
00042 LNK$GW_NFILES::
      .BLKB 2
00044 LNK$GW_NMODULES::
      .BLKB 2
```

00046 LNK\$GW_NPSECTS::
 .BKLB 2
00048 LNK\$GW_NSYMBOLS::
 .BKLB 2
0004A LNK\$GW_LSYMBOLS::
 .BKLB 2
0004C LNK\$GW_NCROSRFS::
 .BKLB 2
0004E LNK\$GW_NUDFSYMS::
 .BKLB 2
00050 LNK\$GW_NLSYMS::
 .BKLB 2
00052 LNK\$GW_NUDFLSYMS::
 .BKLB 2
00054 LNK\$GW_NUDFENV::
 .BKLB 2

ISD\$C_SIZE== 88
HDR\$K_FILLCHR== 255
IHDR\$K_SHR== 2
IHDR\$K_ACTIVOFF== 48
IHDR\$K_SYMDBGOFF== 68
IHDR\$K_IMGIDOFF== 88
IHDR\$K_PATCHOFF== 168
IHDR\$K_MAXLENGTH== 168
HDR\$K_MINFILL== 2
PSC\$M_PSCBITS== 1023
PSC\$M_SHRBITS== 52
RECLNG= OBJRECDESC
OBJREC= OBJRECDESC+4
OBJVEC= OBJRECDESC+4
RECDISPATCH= P.AAA
LNK\$GT_ENVSTRING== P.AAB
LNK\$GT_MODSTRING== P.AAC
LNK\$GT_SYMSTRING== P.AAD
LNK\$GT_PSCSTRING== P.AAE
LNK\$GT_CLUSTRING== P.AAF
LNK\$GT_ENTITY== P.AAG
LNK\$GT_IDENT== P.AAH
LNK\$GT_OBJNAM== P.AAI
LNK\$GT_SYSVER== P.AAJ
 .EXTRN LNK\$CLOSEFILE, LNK\$FILNAMDSC
 .EXTRN LNK\$OPENLIB, LNK\$ALLOFDB
 .EXTRN CRF\$INSRTKEY, CRF\$INSRTREF
 .EXTRN LIB\$INSERT_TREE
 .EXTRN LIB\$LOOKUP_TREE
 .EXTRN LIB\$TRAVERSE_TREE
 .EXTRN LNK\$COMPARE_PSCNOD
 .EXTRN LNK\$SRCPSCDEF, LNK\$BINTIM
 .EXTRN LNK\$ALLOBLK, LNK\$DEALBLK
 .EXTRN LNK\$EXIT, LNK\$FNDPSCNAM
 .EXTRN LNK\$INSERT, LNK\$NXTFIL
 .EXTRN LNK\$NXTREC, LNK\$PROC\$HRIM
 .EXTRN LNK\$PROCSLIB, LNK\$SEARCH
 .EXTRN LNK\$SEARCHLOCAL
 .EXTRN LNK\$ADDIMAGE, LIB\$KEYALRINS
 .EXTRN LNK\$OPENIN, LNK\$BADCCC


```
.EXTRN LINS_BADPSC, LINS_DATMISMCH
.EXTRN LINS_EOMFTL, LINS_ERRORS
.EXTRN LINS_EXCPSC, LINS_FATALERROR
.EXTRN LINS_FORMAT, LINS_GSDTYP
.EXTRN LINS_ENTIDMTCH, LINS_ENTIDMTCHB
.EXTRN LINS_ENTIDMTCHO
.EXTRN LINS_ENTIDMTCHT
.EXTRN LINS_ILLFMLCNT, LINS_ILLNAMELEN
.EXTRN LINS_ILLRECTYP, LINS_ILLRECLN
.EXTRN LINS_MODNAM, LINS_MULDEF
.EXTRN LINS_MULDEFPC, LINS_MULPSC
.EXTRN LINS_MULTFR, LINS_MULSHRPS
.EXTRN LINS_NOEOM, LINS_NOIMGFIL
.EXTRN LINS_NOMODS, LINS_NOPSCS
.EXTRN LINS_NUDFENV, LINS_NUDFSYMS
.EXTRN LINS_NUDFLSYMS, LINS_OVRALI
.EXTRN LINS_PSCALI, LINS_PSCNXR
.EXTRN LINS_RECLNG, LINS_RECTYP
.EXTRN LINS_SEQNCE, LINS_SHRPSCLNG
.EXTRN LINS_STRLVL, LINS_UDFENV
.EXTRN LINS_UDFSYM, LINS_WNERS
.EXTRN LNKSX_MAX_FILENAME_LENGTH
.EXTRN LNKSX_JPILST, LNKSX_FILESLEFT
.EXTRN LNKSX_RAB, LNKSX_SYTBFLMT
.EXTRN LNKSX_VALCTLTB
.EXTRN LNKSX_VERSION, LNKSX_CTLMSK
.EXTRN LNKSX_PSHRNUM, LNKSX_IMGSTA
.EXTRN LNKSX_CLULST, LNKSX_DEFCLU
.EXTRN LNKSX_RECORD, LNKSX_SHRSYMS
.EXTRN LNKSX_SHRIMG, LNKSX_PSCDFLST
.EXTRN LNKSX_MAXSYMSZ
.EXTRN LNKSX_MAXMODSZ
.EXTRN LNKSX_CURCLU, LNKSX_CURFIL
.EXTRN LNKSX_CUROMD, LNKSX_LIBSYM
.EXTRN LNKSX_IMGID, SHRFILTYPE
.EXTRN DEFILETYPE, LIBFILTYPE
```

.PSECT \$CODE\$,NOWRT,2

50

0000 00000 DUMMY:
01 D0 00002
04 00005

.WORD Save nothing
MOVL #1, R0
RET

: 0890
:
:

; Routine Size: 6 bytes, Routine Base: \$CODE\$ + 0000

; 396 0891 1

! ABOVE DECLARATIONS

```

398      0892 1 routine prolnkrec =
399      0893 2 begin
400      0894
401      0895 routine setup_include_list ( omdlstadr ) : novalue =
402      0896 begin
403      0897
404      0898     This routine sets up an include list (in simple linked
405      0899     list form) for modules in a library file spec
406      0900
407      0901 bind      lnkoptrec = .objrec : block [, byte];
408      0902 local    recptr,
409      0903          prevmodule,
410      0904          currentmodule;
411      0905
412      0906 recptr = .objvec + $byteoffset (lnk$st_name)
413      0907          + .lnkoptrec[lnk$w_namlng] ;
414      0908
415      0909 prevmodule = 0;
416      0910 while (.recptr)<0,8> neq 0
417      0911 do begin
418      0912     lnk$alloblk ((.recptr)<0,8>)+5, currentmodule);
419      0913     if .prevmodule eql 0
420      0914     then .omdlstadr = .currentmodule
421      0915     else .prevmodule = .currentmodule;
422      0916
423      0917     prevmodule = .currentmodule;
424      0918     ch$move ((.recptr)<0,8>, .recptr+1, .currentmodule+5);
425      0919     (.currentmodule+4)<0,8> = (.recptr)<0,8>;
426      0920     .currentmodule = 0;
427      0921     recptr = .recptr + (.recptr)<0,8> + 1;
428      0922 end;
429      0923 return;
end;
```

```

                                07FC 00000 SETUP_INCLUDE_LIST:
                                .WORD      Save R2,R3,R4,R5,R6,R7,R8,R9,R10
0A 00000000' EF 9E 00002 MOVAB OBJREC, R10
5E 04 C2 00009 SUBL2 #4, SP
50 6A D0 0000C MOVL OBJREC, R0
56 04 A0 3C 0000F MOVZWL 4(R0), R6
56 6A C0 00013 ADDL2 OBJREC, R6
56 06 C0 00016 ADDL2 #6, RECPTR
58 59 D4 00019 CLRL PREVMODULE
66 9A 0001B 1$: MOVZBL (RECPTR), R8
32 13 0001E BEQL 4$
5E DD 00020 PUSHL SP
05 A8 9F 00022 PUSHAB 5(R8)
00000000G 00 02 FB 00025 CALLS #2, LNK$ALLOBLK
57 6E D0 0002C MOVL CURRENTMODULE, R7
59 D5 0002F TSTL PREVMODULE
06 12 00031 BNEQ 2$
04 BC 57 D0 00033 MOVL R7, @OMDLSTADR
03 11 00037 BRB 3$
69 57 D0 00039 2$: MOVL R7, (PREVMODULE)
```


LNK_OBJPASS1
V04=000

K 12
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 14
(4)

05	A7	01	59	57	D0	0003C	3\$:	MOVL	R7, PREVMODULE	:	0916
		04	A6	58	28	0003F		MOVC3	R8, 1(RECPTR), 5(R7)	:	0917
			A7	58	90	00045		MOVB	R8, 4(R7)	:	0918
			56	67	D4	00049		CLRL	(R7)	:	0919
				01	A846	9E	00048	MOVAB	1(R8)[RECPTR], RECPTR	:	0920
				C9	11	00050		BRB	1\$	:	0909
					04	00052	4\$:	RET		:	0923

; Routine Size: 83 bytes, Routine Base: \$CODE\$ + 0006

```

431      0924 2 local
432      0925      lastfdb      : ref block [,byte],
433      0926      auxfnb      : ref block [,byte],
434      0927      input_fab   : block [fab$C_bln, byte],
435      0928      errorcode,
436      0929      next_fdb,
437      0930      filnamadr,
438      0931      fildesblk      : ref block [,byte];
439      0932  bind
440      0933      lnkoptrec = .objrec : block [, byte];
441      0934  !
442      0935  ! PROCESS LINKER OPTIONS RECORD
443      0936  !
444      0937  if .lnkoptrec [lnk$b_lnktyp] gtr lnk$c_maxrectyp      ! Valid subrecord type?
445      0938  or .lnkoptrec [lnk$w_namlng] gtr nam$c_maxrss      ! Reasonable-looking name length?
446      0939  then return false;
447      0940
448      0941  lnk$alloblk (.lnkoptrec [lnk$w_namlng], filnamadr);      ! allocate storage for filename
449      0942  ch$move (.lnkoptrec [lnk$w_namlng], lnkoptrec [lnk$t_name], .filnamadr);
450      0943  lnk$allofdb (fildesblk);
451      0944  auxfnb = fildesblk [fdb$t_auxfnb];
452      0945  $fab_init (fab = input_fab      ! initialize the fab
453      0946      , fac = get      ! for input
454      0947      , fop = sqo
455      0948      , shr = (upi,get,put)
456      0949      , fns = .lnkoptrec [lnk$w_namlng]
457      0950      , fna = .filnamadr
458      0951      , nam = .auxfnb
459      0952      , dns = 4
460      0953      , dna = ch$ptr (libfiltype)
461      0954      );
462      0955  fildesblk [fdb$w_usrnamlen] = .lnkoptrec [lnk$w_namlng];      ! fill in file name
463      0956  fildesblk [fdb$l_usrnamadr] = .filnamadr;
464      0957
465      0958  select .lnkoptrec [lnk$b_lnktyp] of      ! determine the file type
466      0959  set
467      0960  [lnk$c_olb] : fildesblk [fdb$v_libr] = true;      ! all of these are libraries
468      0961  [lnk$c_oli] : fildesblk [fdb$v_libsrch] = true;      ! olb to be searched
469      0962  [lnk$c_shr] : fildesblk [fdb$v_imglib] = true;      ! shareable image LIBRARY
470      0963  [lnk$c_oli] : begin      ! object lib with include list
471      0964      fildesblk [fdb$v_libextr] = true;
472      0965      fildesblk [fdb$v_libsrch] = .lnkoptrec [lnk$v_libsrch];
473      0966      setup_include_list (fildesblk [fdb$l_omdlst]);      ! put modules in linked list
474      0967  end;
475      0968  [lnk$c_obj] : begin      ! regular object module
476      0969      fildesblk [fdb$v_selser] = .lnkoptrec [lnk$v_selser];
477      0970      input_fab [fab$l_dna] = ch$ptr (defiletype);
478      0971  end;
479      0972  [lnk$c_sha] : begin      ! shareable image
480      0973      fildesblk [fdb$v_shr] = true;
481      0974      input_fab [fab$l_dna] = ch$ptr (shrfiletype);
482      0975  end;
483      0976  tes;
484      0977
485      0978
486      0979
487      0980 2 if .fildesblk[fdb$v_libr] or .fildesblk[fdb$v_imglib]      ! if a library file
```



```
488 0981 3 then begin
489 0982 3   $sparse (fab=input_fab);
490 0983 4   if (errorcode = $search (fab=input_fab))
491 0984 4   then begin
492 0985 4       local      savcurfil;
493 0986 4       savcurfil   = .lnk$gl_curfil;
494 0987 4       lnk$gl_curfil = .fildesblk;
495 0988 4       errorcode   = lnk$openlib (.auxfnb);
496 0989 4       lnk$gl_curfil = .savcurfil;
497 0990 4       end;
498 0991 4   end
499 0992 4 else begin
500 0993 4   $getjpi (itmlst=lnk$gt_jpilst);
501 0994 4   if .lnk$gl_filesleft leq 3 then lnk$closefile ();
502 0995 4   errorcode = $open (fab=input_fab);
503 0996 4   fildesblk [fdb$w_ifi] = .input_fab [fab$w_ifi];
504 0997 4   end;
505 0998 4
506 0999 4
507 1000 4 if not .errorcode
508 1001 4 then signal_stop (lin$openin, 1
509 1002 4   ,lnk$filnamdsc (input_fab), .errorcode
510 1003 4   ,input_fab [fab$l_stv]
511 1004 4   );
512 1005 4
513 1006 4 if .fildesblk [fdb$v_shr]
514 1007 4 then begin
515 1008 4   local      moduledsc : vector [2, long];
516 1009 4
517 1010 4   moduledsc [0] = .auxfnb [nam$b_name];
518 1011 4   moduledsc [1] = .auxfnb [nam$l_name];
519 1012 4   lnk$addimage (moduledsc);
520 1013 4   end
521 1014 4 else begin
522 1015 4   if .last_lnkoft_fdb eql 0
523 1016 4   then last_lnkoft_fdb = .lnk$gl_curfil;
524 1017 4
525 1018 4   if .lnk$gl_curclu[clu$l_lstfdb] eql .last_lnkoft_fdb
526 1019 4   then lnk$gl_curclu[clu$l_lstfdb] = .fildesblk;
527 1020 4
528 1021 4   next_fdb
529 1022 4   last_lnkoft_fdb [fdb$l_nxtfdb] = .fildesblk;
530 1023 4   fildesblk [fdb$l_nxtfdb]
531 1024 4   last_lnkoft_fdb
532 1025 4   = .fildesblk;
533 1026 4   end;
534 1027 4 return true;
534 1027 4 end;
```

! then fill in name block enough
! to do a library open by name block
! set ptr to current fdb for the temporary b
! of lnk\$openlib
! must now restore it
! not a library file
! close a file if needed
! and try again
! save ifi
! if either of the opens was a failure,
! then signal and quit
! Shareable image...?
! Shareable image spec: set up name
! and create its very own cluster
! Not a shareable image spec
! if first LNKOPT rec for this file
! set last_lnkoft_fdb to current file
! If last fdb is last in current cluster
! then must update last cluster fdb
! Save ptr to next fdb
! Point current to this new fdb
! Point new fdb to last
! Update fdb of last LNKOPT file

.EXTRN SYSSPARSE, SYSSSEARCH
.EXTRN SYSSGETJPI, SYSSOPEN

OFFC 00000 PROLNKREC:

5B 00000000G	00	9E 00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	: 0892
5A 00000000'	EF	9E 00009	MOVAB	LNK\$GL_CURFIL, R11	:
			MOVAB	LAST_LNKOPT_FDB, R10	:

0050	8F	00	00FF	8F	04	01A2	31	0002A	1\$:	MOVAB	-96(SP), SP	0933
						03	1B	00028	2\$:	MOVL	OBJREC, R7	0937
						06	1A	00020		MOVZBL	1(R7), R9	
						07	B1	00022		CMPB	R9, #4	
						08	03	00028		BGTRU	1\$	
						09	03	00028		CMPW	4(R7), #255	0938
						0A	03	00028		BLEQU	2\$	
						0B	03	00028		BRW	20\$	
						0C	03	00028		PUSHL	SP	0941
						0D	03	00028		MOVZWL	4(R7), -(SP)	
						0E	03	00028		CALLS	#2, LNK\$ALLOBLK	
						0F	03	00028		MOVC3	4(R7), 6(R7), @FILNAMADR	0942
						10	03	00028		PUSHAB	FILDESBLK	0943
						11	03	00028		CALLS	#1, LNK\$ALLOFDB	
						12	03	00028		MOVL	FILDESBLK, R8	0944
						13	03	00028		MOVAB	38(R8), AUXFNB	
						14	03	00028		MOVC5	#0, (SP), #0, #80, \$RMS_PTR	0954
						15	03	00028				
						16	03	00028		MOVW	#20483, \$RMS_PTR	
						17	03	00028		MOVZBL	#64, \$RMS_PTR+4	
						18	03	00028		MOVW	#17154, \$RMS_PTR+22	
						19	03	00028		MOVB	#2, \$RMS_PTR+31	
						1A	03	00028		MOVL	AUXFNB, \$RMS_PTR+40	
						1B	03	00028		MOVL	FILNAMADR, \$RMS_PTR+44	
						1C	03	00028		MOVAB	LIBFILTYPE, \$RMS_PTR+48	
						1D	03	00028		MOVB	4(R7), \$RMS_PTR+52	
						1E	03	00028		MOVB	#4, \$RMS_PTR+53	
						1F	03	00028		MOVW	4(R7), 12(R8)	0955
						20	03	00028		MOVL	FILNAMADR, 16(R8)	0956
						21	03	00028		TSTL	R9	0960
						22	03	00028		BEQL	3\$	
						23	03	00028		CMPB	R9, #2	
						24	03	00028		BNEQ	4\$	
						25	03	00028		BISB2	#2, 10(R8)	0961
						26	03	00028		TSTL	R9	0962
						27	03	00028		BNEQ	5\$	
						28	03	00028		BISB2	#128, 10(R8)	
						29	03	00028		CMPB	R9, #1	0963
						2A	03	00028		BNEQ	6\$	
						2B	03	00028		BISB2	#1, 11(R8)	
						2C	03	00028		CMPB	R9, #2	0964
						2D	03	00028		BNEQ	7\$	
						2E	03	00028		BISB2	#64, 10(R8)	0965
						2F	03	00028		EXTZV	#1, #1, 2(R7), R0	0966
						30	03	00028		INSV	R0, #7, #1, 10(R8)	
						31	03	00028		PUSHAB	4(R8)	0967
						32	03	00028		CALLS	#1, SETUP_INCLUDE_LIST	
						33	03	00028		CMPB	R9, #3	0969
						34	03	00028		BNEQ	8\$	
						35	03	00028		INSV	2(R7), #3, #1, 10(R8)	0970
						36	03	00028		MOVAB	DEFILTYPE, INPUT_FAB+48	0971
						37	03	00028		CMPB	R9, #4	0973
						38	03	00028		BNEQ	9\$	
						39	03	00028		BISB2	#4, 10(R8)	0974
						3A	03	00028		MOVAB	SHRFILTYPE, INPUT_FAB+48	0975
						3B	03	00028		BBS	#1, 10(R8), 10\$	0980
						3C	03	00028		BLBC	11(R8), 11\$	

00000000G	00	10	AE	9F	000FE	10\$:	PUSHAB	INPUT_FAB	0982
			01	FB	00101		CALLS	#1, SYSSPARSE	
00000000G	00	10	AE	9F	00108		PUSHAB	INPUT_FAB	0983
	53		01	FB	0010B		CALLS	#1, SYSSSEARCH	
	51		50	D0	00112		MOVL	R0, ERRORCODE	
	52		53	E9	00115		BLBC	ERRORCODE, 14\$	
	6B		6B	D0	00118		MOVL	LNK\$GL_CURFIL, SAVCURFIL	0986
			58	D0	0011B		MOVL	R8, LNK\$GL_CURFIL	0987
			56	DD	0011E		PUSHL	AUXFNB	0988
00000000G	00		01	FB	00120		CALLS	#1, LNK\$OPENLIB	
	53		50	D0	00127		MOVL	R0, ERRORCODE	
	6B		52	D0	0012A		MOVL	SAVCURFIL, LNK\$GL_CURFIL	0989
			37	11	0012D		BRB	13\$	0980
			7E	7C	0012F	11\$:	CLRQ	-(SP)	0993
			7E	D4	00131		CLRL	-(SP)	
		00000000G	00	9F	00133		PUSHAB	LNK\$GT_JPILST	
			7E	7C	00139		CLRQ	-(SP)	
			7E	D4	0013B		CLRL	-(SP)	
00000000G	00		07	FB	0013D		CALLS	#7, SYSSGETJPI	
	03	00000000G	00	D1	00144		CMPL	LNK\$GL_FILESLEFT, #3	0994
			07	14	0014B		BGTR	12\$	
00000000G	00		00	FB	0014D		CALLS	#0, LNK\$CLOSEFILE	
		10	AE	9F	00154	12\$:	PUSHAB	INPUT_FAB	0995
00000000G	00		01	FB	00157		CALLS	#1, SYSSOPEN	
	53		50	D0	0015E		MOVL	R0, ERRORCODE	
24	A8	12	AE	B0	00161		MOVW	INPUT_FAB+2, 36(R8)	0996
	20		53	E8	00166	13\$:	BLBS	ERRORCODE, 15\$	1000
		1C	AE	DD	00169	14\$:	PUSHL	INPUT_FAB+12	1003
			53	DD	0016C		PUSHL	ERRORCODE	1002
		18	AE	9F	0016E		PUSHAB	INPUT_FAB	
00000000G	00		01	FB	00171		CALLS	#1, LNK\$FILNAMDSC	
			50	DD	00178		PUSHL	R0	
			01	DD	0017A		PUSHL	#1	1001
		00000000G	8F	DD	0017C		PUSHL	#LINS OPENIN	
00000000G	00		05	FB	00182		CALLS	#5, LIB\$STOP	
	0A	A8	02	E1	00189	15\$:	BBC	#2, 10(R8), 16\$	1006
	08	AE	A6	9A	0018E		MOVZBL	59(AUXFNB), MODULEDSC	1010
	0C	AE	A6	D0	00193		MOVL	76(AUXFNB), MODULEDSC+4	1011
		08	AE	9F	00198		PUSHAB	MODULEDSC	1012
00000000G	00		01	FB	0019B		CALLS	#1, LNK\$ADDIMAGE	
			27	11	001A2		BRB	19\$	1006
			6A	D5	001A4	16\$:	TSTL	LAST_LNKOPT_FDB	1015
			03	12	001A6		BNEQ	17\$	
	6A		6B	D0	001A8		MOVL	LNK\$GL_CURFIL, LAST_LNKOPT_FDB	1016
	50	00000000G	00	D0	001AB	17\$:	MOVL	LNK\$GL_CURCLU, R0	1018
	51		6A	D0	001B2		MOVL	LAST_LNKOPT_FDB, R1	
	51	0C	A0	D1	001B5		CMPL	12(R0), R1	
			04	12	001B9		BNEQ	18\$	
	0C	A0	58	D0	001BB		MOVL	R8, 12(R0)	1019
	50		61	D0	001BF	18\$:	MOVL	(R1), NEXT_FDB	1021
	61		58	D0	001C2		MOVL	R8, (R1)	1022
	68		50	D0	001C5		MOVL	NEXT_FDB, (R8)	1023
	6A		58	D0	001C8		MOVL	R8, LAST_LNKOPT_FDB	1024
	50		01	D0	001CB	19\$:	MOVL	#1, R0	1026
				04	001CE		RET		
			50	D4	001CF	20\$:	CLRL	R0	1027
				04	001D1		RET		

LNK_OBJPASS1
V04=000

C 13
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 19
(5)

; Routine Size: 466 bytes, Routine Base: \$CODE\$ + 0059


```

: 536      1028 1 routine checkpddentry (pdd) =
: 537      1029 2   begin
: 538      1030 3   !
: 539      1031 4   ! ROUTINE CALLED BY LIB$TRAVERSE_TREE FOR EACH PSECT DEFINED BY OPTION
: 540      1032 5   !
: 541      1033 6   map
: 542      1034 7     pdd : ref block [, byte];
: 543      1035 8   local
: 544      1036 9     psctdesc : ref block [, byte];
: 545      1037 10
: 546      1038 11 if (.pdd [pdd$b_namlng] and %x'80') eql 0           ! IF NOT DEFINED YET
: 547      1039 12 then begin
: 548      1040 13   lnk$findpscnam (pdd [pdd$b_namlng], .pdd [pdd$w_flags]           ! THEN DO SO NOW
: 549      1041 14   ,((.pdd [pdd$w_flags] and gps$m_gbl) neq 0)
: 550      1042 15   ,psctdesc
: 551      1043 16   );
: 552      1044 17   psctdesc [psc$w_flags] = .pdd [pdd$w_flags];           ! SET THE FLAGS
: 553      1045 18
: 554      1046 19   if (psctdesc [psc$b_align] = .pdd [pdd$b_align]) eql %x'FF' ! AND THE ALIGNMENT
: 555      1047 20   then psctdesc [psc$b_align] = 0;
: 556      1048 21 end;
: 557      1049 22
: 558      1050 23 return true
: 559      1051 24 end;
```

0004 00000 CHECKPDDENTRY:									
									1028
	5E		04	C2	00002				
	52		04	AC	D0	00005			
			0F	A2	95	00009			1038
				32	19	0000C			
				5E	DD	0000E			1040
				7E	D4	00010			1041
02	0A	A2		04	E1	00012			
				6E	D6	00017			
	7E		0A	A2	3C	00019	1\$:		1040
			0F	A2	9F	0001D			
00000000G	00			04	FB	00020			
	50			6E	D0	00027			1044
	0A	A0	0A	A2	B0	0002A			
	52		0E	A2	9A	0002F			1046
	2C	A0		52	90	00033			
	FF	8F		52	91	00037			
				03	12	0003B			
			2C	A0	94	0003D			1047
	50			01	D0	00040	2\$:		1050
				04	00043				1051

Save R2
#4, SP
PDD, R2
15(R2)
2\$
SP
-(SP)
#4, 10(R2), 1\$
(SP)
10(R2), -(SP)
15(R2)
#4, LNK\$FNDPSCNAM
PSCTDESC, R0
10(R2), 10(R0)
14(R2), R2
R2, 44(R0)
R2, #255
2\$
44(R0)
#1, R0

; Routine Size: 68 bytes, Routine Base: \$CODE\$ + 022B

```

561 1052 1 global routine lnk$insudfsym (symboladdr) : novalue =
562 1053 2 begin
563 1054 2 :
564 1055 2 : THIS ROUTINE INSERTS AN UNDEFINED SYMBOL INTO THE LINKED LIST
565 1056 2 : OF UNDEFINED SYMBOLS.
566 1057 2 :
567 1058 2 map
568 1059 2 symboladdr : ref block [, byte]; ! REALLY A POINTER
569 1060 2 builtin
570 1061 2 insque;
571 1062 2 local
572 1063 2 ch_result,
573 1064 2 listhead : ref vector [, long],
574 1065 2 nxtsyment : ref block [, byte]; ! POINTER TO SYMBOL VALUE BLOCK
575 1066 2 bind
576 1067 2 symbolname = .symboladdr ! POINT TO NAME PART
577 1068 2 - .symboladdr [sym$b_namlng]
578 1069 2 - snb$c_fxdlen : block [, byte];
579 1070 2 :
580 1071 2 :
581 1072 2 : FIND THE SPOT TO INSERT THE NEW SYMBOL IN THE UNDEFINED LIST
582 1073 2 :
583 1074 2 listhead = lnk$gl_udflst;
584 1075 2 if .lclsymgsd then listhead = lnk$gl_udflsy;
585 1076 2 :
586 1077 2 nxtsyment = .listhead; ! SCAN THE UNDEFINED LIST
587 1078 2 :
588 1079 2 while (nxtsyment = .nxtsyment [sym$l_udflink]) neq .listhead
589 1080 2 do begin
590 1081 2 bind ! POINT TO NAME PART
591 1082 2 nxtsymnam = .nxtsyment - .nxtsyment [sym$b_namlng]
592 1083 2 - snb$c_fxdlen : block [, byte];
593 1084 2 :
594 1085 2 if (ch_result = ch$compare (.nxtsymnam [snb$b_namlng] ! IF GREATER
595 1086 2 , ntxtsymnam [snb$t_name]
596 1087 2 , .symbolname [snb$b_namlng]
597 1088 2 , symbolname [snb$t_name]
598 1089 2 ) gtr 0
599 1090 2 then exitloop ! THEN WE ARE ALL DONE
600 1091 2 else if .ch_result eql 0 ! IF EQUAL
601 1092 2 then return; ! THEN ALREADY IN LIST, SO RETURN
602 1093 2 end;
603 1094 2 :
604 1095 2 :
605 1096 2 insque (symboladdr [sym$l_udflink], .nxtsyment [sym$l_udfblink]); ! INSERT IN UNDEFINED LIST
606 1097 2 :
607 1098 2 if not .symboladdr [sym$v_weak] ! IF NOT A WEAK REFERENCE
608 1099 2 then if .lclsymgsd
609 1100 2 then lnk$gw_nudflsyms = .lnk$gw_nudflsyms + 1 ! THEN COUNT AS UNDEFINED
610 1101 2 else lnk$gw_nudflsyms = .lnk$gw_nudflsyms + 1;
611 1102 2 :
612 1103 2 return;
613 1104 2 end;
```


			OFFC	00000		.ENTRY	LNK\$INSUDFSYM, Save R2,R3,R4,R5,R6,R7,R8,-	
	5B	00000000'	EF	9E	00002	MOVAB	R9,R10,R11	1052
	5A	00000000'	EF	9E	00009	MOVAB	LCLSYMGS, R11	
	57	04	AC	D0	00010	MOVAB	LNK\$GL_UDFLST, R10	
	50	0F	A7	9A	00014	MOVL	SYMBOLADDR, R7	1067
50	57		50	C3	00018	MOVZBL	15(R7), R0	1068
	56	FB	A0	9E	0001C	SUBL3	R0, R7, R0	
	59		6A	9E	00020	MOVAB	-5(R0), R6	1069
	04		6B	E9	00023	MOVAB	LNK\$GL_UDFLST, LISTHEAD	1074
	59	14	AA	9E	00026	BLBC	LCLSYMGS, 1\$	1075
	54		59	D0	0002A	MOVAB	LNK\$GL_UDFLSY, LISTHEAD	
	54		64	D0	0002D	MOVL	LISTHEAD, NXTSYMENT	1077
	59		54	D1	00030	MOVL	(NXTSYMENT), NXTSYMENT	1079
			2B	13	00033	CMPL	NXTSYMENT, LISTHEAD	
	50	0F	A4	9A	00035	BEQL	4\$	
50	54		50	C3	00039	MOVZBL	15(NXTSYMENT), R0	1082
	50		05	C2	0003D	SUBL3	R0, NXTSYMENT, R0	
	52	04	A0	9A	00040	SUBL2	#5, R0	1083
	51	04	A6	9A	00044	MOVZBL	4(R0), R2	1085
	55		01	D0	00048	MOVZBL	4(R6), R1	1087
51	00	05	A0	2D	0004B	MOVL	#1, R5	1088
		05	A6		00051	CMPC5	R2, 5(R0), #0, R1, 5(R6)	
			03	1A	00053	BGTRU	3\$	
	55		01	D9	00055	SBWC	#1, R5	
	58		55	D0	00058	MOVL	R5, CH_RESULT	
			03	14	0005B	BGTR	4\$	1090
			CE	12	0005D	BNEQ	2\$	1092
				04	0005F	RET		1093
	04	B4	67	0E	00060	INSQUE	(R7), @4(NXTSYMENT)	1096
	50	04	AC	D0	00064	MOVL	SYMBOLADDR, R0	1098
	0A	0A	A0	E8	00068	BLBS	10(R0), 6\$	
	04		6B	E9	0006C	BLBC	LCLSYMGS, 5\$	1099
		4A	AA	B6	0006F	INCW	LNK\$GW_NUDFLSYMS	1100
				04	00072	RET		
		46	AA	B6	00073	INCW	LNK\$GW_NUDFSYMS	1101
			04	00076	6\$:	RET		1104

; Routine Size: 119 bytes, Routine Base: \$CODE\$ + 026F

; 614 1105 1

```

616 1106 1 routine seqchk =
617 1107 1
618 1108 1 ROUTINE WHICH VALIDATES THAT RECORDS ARE IN CORRECT SEQUENCE.
619 1109 1 RETURNS VALUE FALSE IF NOT, TRUE OTHERWISE. ALSO TURNS OFF THE NO
620 1110 1 BINARY FLAG WHEN A TIR RECORD IS SEEN.
621 1111 1
622 1112 2 begin
623 1113 2 bind
624 1114 2     hdrsubtyp = objrec [obj$b_subtyp] : byte;
625 1115 2
626 1116 2 if .currenttyp eql obj$c_hdr
627 1117 2 then if .hdrsubtyp eql obj$c_hdr_mhd
628 1118 2     then
629 1119 2         if (.lastrectyp eql obj$c_eom)
630 1120 2         or (.lastrectyp eql obj$c_eomw)
631 1121 2         then begin
632 1122 2             mhdseen = true;
633 1123 2             lnmseen = false;
634 1124 2             return true
635 1125 2         end
636 1126 2         else begin
637 1127 2             signal (lin$seqnce, 2
638 1128 2                 ,obmodesc [omd$b_namlng]
639 1129 2                 ,lnk$gl_curfil [fdb$q_filename]
640 1130 2                 );
641 1131 2             return false;
642 1132 2         end
643 1133 2     else if .mhdseen
644 1134 2     then (if .hdrsubtyp eql obj$c_hdr_lnm
645 1135 2         then lnmseen = true;
646 1136 2         return true
647 1137 2         )
648 1138 2     else begin
649 1139 2         signal (lin$seqnce, 2
650 1140 2             ,obmodesc [omd$b_namlng]
651 1141 2             ,lnk$gl_curfil [fdb$q_filename]
652 1142 2             );
653 1143 2         return false;
654 1144 2         end
655 1145 2     else if .mhdseen and .lnmseen
656 1146 2     then begin
657 1147 2         if (.currenttyp eql obj$c_eom)
658 1148 2         or (.currenttyp eql obj$c_eomw)
659 1149 2         then mhdseen = false
660 1150 2         else if .currenttyp eql obj$c_tir
661 1151 2         then obmodesc [omd$v_nobin] = false;
662 1152 2
663 1153 2         return true;
664 1154 2         end
665 1155 2     else begin
666 1156 2         signal (lin$seqnce, 2
667 1157 2             ,obmodesc [omd$b_namlng]
668 1158 2             ,lnk$gl_curfil [fdb$q_filename]
669 1159 2             );
670 1160 2         return false;
671 1161 2         end;
672 1162 1 end;
```

! IF THIS RECORD IS A HEADER
! AND IT IS THE MAIN MODULE HEADER
! THEN WE HAVE VALID SEQUENCE
! IF AND ONLY IF THE PREVIOUS WAS END OF MOD

! IS THIS CASE SET MHD RECORD SEEN AND RETUR

! ELSE REPORT ERROR

! IF SOME OTHER KIND OF HEADER
! WE MUST HAVE SEEN A MAIN HEADER

! ELSE REPORT ERROR

! IF WE HAVE SEEN A MAIN HEADER
! THEN TURN OFF FLAG ON END OF MODULE.
! SEQUENCE ERROR IF HAVE NOT SEEN

! MAIN HEADER AND THIS IS NOT ONE.

! ELSE REPORT ERROR

51	11	52	00000000'	0004	00000	SEQCHK:	.WORD	Save R2	:	1106
		A2		EF	9E		MOVAB	MHDSEEN, R2	:	1114
		50	03	01	C1		ADDL3	#1, OBJREC, R1	:	1116
				A2	9A		MOVZBL	CURRECTYP, R0	:	
				23	12		BNEQ	3\$	:	
				61	95		TSTB	(R1)	:	1117
				11	12		BNEQ	2\$	:	
		03	02	A2	91		CMPB	LASTRECTYP, #3	:	1119
		07	02	06	13		BEQL	1\$	:	
				A2	91		CMPB	LASTRECTYP, #7	:	1120
		62		35	12		BNEQ	7\$	:	
				01	B0	1\$:	MOVW	#1, MHDSEEN	:	1122
		2D		2C	11		BRB	6\$	:	1124
		01		62	E9	2\$:	BLBC	MHDSEEN, 7\$	:	1133
				61	91		CMPB	(R1), #1	:	1134
	01	A2		24	12		BNEQ	6\$	:	
				01	90		MOVB	#1, LNMSEEN	:	1135
		1F		1E	11		BRB	6\$	:	1136
		1B		62	E9	3\$:	BLBC	MHDSEEN, 7\$	:	1145
		03	01	A2	E9		BLBC	LNMEEN, 7\$	:	
				50	91		CMPB	R0, #3	:	1147
		07		05	13		BEQL	4\$	:	
				50	91		CMPB	R0, #7	:	1148
				04	12		BNEQ	5\$	:	
				62	94	4\$:	CLRB	MHDSEEN	:	1149
		02		09	11		BRB	6\$	:	
				50	91	5\$:	CMPB	R0, #2	:	1150
				04	12		BNEQ	6\$	:	
	55	A2		02	8A		BICB2	#2, OBMODESC+20	:	1151
		50		01	D0	6\$:	MOVL	#1, R0	:	1153
					04		RET		:	
7E	00000000G	00		14	C1	7\$:	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	:	1158
			69	A2	9F		PUSHAB	OBMODESC+40	:	1157
				02	DD		PUSHL	#2	:	1158
				8F	DD		PUSHL	#LINS_SEQNCE	:	
00000000G	00		00000000G	04	FB		CALLS	#4, LIB\$SIGNAL	:	
				50	D4		CLRL	R0	:	1162
				04	00075		RET		:	

; Routine Size: 118 bytes, Routine Base: \$CODE\$ + 02E6

; 673 1163 1

```

675 1164 1 routine prohdr =
676 1165 1 ++
677 1166 1 PROCESS MODULE HEADER RECORDS AS FOLLOWS:
678 1167 1 (1) VALIDATE SEQUENCE
679 1168 1 (2) IGNORE ALL BUT MAIN MODULE HEADERS
680 1169 1 (3) VERIFY STRUCTURE LEVEL IS LESS THAN
681 1170 1 OR EQUAL TO OBJ$C_STRLVL
682 1171 1 (4) VERIFY MAXIMUM RECORD LENGTH
683 1172 1 PARAMETER IS LESS THAN OR EQUAL TO
684 1173 1 OBJ$C_MAXRECSIZ
685 1174 1 (5) RECORD MAXIMUM RECORD LENGTH PARAMETER
686 1175 1 FOR CHECKING SUBSEQUENT RECORDS
687 1176 1 (6) CHECK MODULE TITLE > 0 AND LESS THAN OR
688 1177 1 EQUAL TO SYM$C_MAXLNG CHARACTERS
689 1178 1 (7) ALLOCATE AN OBJECT MODULE DESCRIPTOR
690 1179 1 AND PUT ON END OF THE LIST (WHOSE
691 1180 1 HEAD IS FILE DESCRIPTOR BLOCK).
692 1181 1 (8) COPY IN THE MODULE TITLE
693 1182 1 (9) INITIALIZE FLAGS AND P-SECTION COUNT IN
694 1183 1 THE OBJECT MODULE DESCRIPTOR
695 1184 1 --
696 1185 2 begin
697 1186 2 bind
698 1187 2 mhdrec = .objrec : block [, byte] ;
699 1188 2 local
700 1189 2 module_max_length ;
701 1190 2
702 1191 2 if not seqchk () then return false ; ! VALIDATE CORRECT SEQUENCE OF RECORD
703 1192 2
704 1193 2 if .objrec [obj$b_subtyp] neq obj$c_hdr_mhd then return true ; ! IGNORE ALL HEADERS EXCEPT MAIN ONES
705 1194 2
706 1195 2 lnk$gw_nmodules = .lnk$gw_nmodules + 1 ; ! COUNT THIS MODULE
707 1196 2
708 1197 2 if .mhdrec [mhd$b_strlvl] gtru obj$c_strlvl ! COMPARE ITS OBJ FORMAT
709 1198 2 then begin ! LEVEL AND IF BEYOND THIS AND GIVE UP
710 1199 2 signal (lin$_strlvl, 4
711 1200 2 , .mhdrec [mhd$b_strlvl], obj$c_strlvl
712 1201 2 , mhdrec [mhd$b_namlng], lnk$gl_curfil [fdb$q_filename]
713 1202 2 ) ;
714 1203 2 return false ;
715 1204 2 end ;
716 1205 2
717 1206 2 if (maxreclng = .mhdrec [mhd$w_recsiz]) gtru obj$c_maxrecsiz ! COMPARE MAX WITH MAX ALLOWED
718 1207 2 then begin ! AND IF GREATER
719 1208 2 signal (lin$_illreclen, 3 ! ISSUE ERROR MESSAGE
720 1209 2 , .maxreclng, mhdrec [mhd$b_namlng]
721 1210 2 , lnk$gl_curfil [fdb$q_filename]
722 1211 2 ) ;
723 1212 2 return false ;
724 1213 2 end ;
725 1214 2
726 1215 2 module_max_length = lnk$k_max_filename_length ; ! SETUP FOR ERROR REPORT
727 1216 2
728 1217 2 if .mhdrec [mhd$b_namlng] eql 0 ! IN CASE mhd$b_namlng = 0
729 1218 2 or begin ! CHECK MODULE NAME LENGTH
730 1219 2 if .lnk$gl_curclu [clu$v_shring] ! IS WITHIN LEGAL LIMITS...
731 1220 2 then if .mhdrec [mhd$b_namlng] gtru lnk$k_max_filename_length ! 1 to 39 for shareable images...
```



```
732 1221 4      then begin
733 1222 4          module_max_length = lnk$$_max_filename_length ;
734 1223 4          true
735 1224 4          end
736 1225 3      else false
737 1226 3      else if .mhdrec [mhd$b_namlng] gtru sym$_maxlng      ! 1 TO 31 FOR OBJECTS...
738 1227 4          then begin
739 1228 4              module_max_length = sym$_maxlng ;
740 1229 4              true
741 1230 4              end
742 1231 3      else false
743 1232 3      end
744 1233 2      then
745 1234 2          begin
746 1235 2              signal ( lnk$_illnamelen, 6, lnk$_modstring      ! AND IF NOT ISSUE ERROR
747 1236 2                  , mhdrec [mhd$b_namlng], .mhdrec [mhd$b_namlng]      ! MESSAGE AND QUIT
748 1237 2                  , .module_max_length, mhdrec [mhd$b_namlng]
749 1238 2                  , lnk$_curfil [fdb$_filename]
750 1239 2              ) ;
751 1240 2          return false ;
752 1241 2          end ;
753 1242 2
754 1243 2      obmodesc [omd$_ownfdb] = .lnk$_curfil ;      ! SET POINTER TO OWNING FDB
755 1244 2      obmodesc [omd$_modvbn] = .lnk$_rab [rab$_rfa0] ;      ! THIS IS NOW THE LAST. CAPTURE THE
756 1245 2      obmodesc [omd$_bytoff] = .lnk$_rab [rab$_rfa4] ;      ! MODULE'S RFA FOR LATER
757 1246 2      obmodesc [omd$_namlng] = .mhdrec [mhd$b_namlng] ;      ! COPY THE STRING LENGTH
758 1247 2
759 1248 2      ch$copy (.mhdrec [mhd$b_namlng], mhdrec [mhd$_name], 0      ! AND THE STRING ZERO FILLED
760 1249 2          , sym$_maxlng, obmodesc [omd$_name]) ;      ! INTO THE DESCRIPTOR
761 1250 2
762 1251 2      obmodesc [omd$_hipsct] = -1 ;      ! SET HIGHEST POSSIBLE P-SECTION NUMBER
763 1252 2      obmodesc [omd$_hienv] = -1 ;      ! SET HIGHEST POSSIBLE ENVIRONMENT NUMBER
764 1253 2      obmodesc [omd$_flags] = omd$_nopsct or omd$_nobin      ! FLAG NO P-SECTS AND NO BINARY YET
765 1254 2          or omd$_noenv ;      ! AND NO ENVIRONMENTS DEFINED
766 1255 2      obmodesc [omd$_shring] = .lnk$_curfil [fdb$_shr] ;      ! EXTRACT THE SHAREABLE IMAGE ATTRIB
767 1256 2          FROM FILE DESCRIPTOR
768 1257 2      obmodesc [omd$_selser] = .lnk$_curfil [fdb$_selser] ;      ! AS WELL AS THE SELECTIVE SEARCH ATTRIBUTE
769 1258 2      obmodesc [omd$_debuger] = .lnk$_curfil [fdb$_debuger] ;      ! COPY DEBUGGER MODULE FLAG
770 1259 2      obmodesc [omd$_omdnum] = .lnk$_nmodules ;      ! SET MODULE NUMBER INTO DESCRIPTOR
771 1260 2
772 1261 2      if not .lnk$_ctlmsk [lnk$_intfil]      ! IF THIS IS NOT STARLET
773 1262 2      then obmodesc [omd$_mapmod] = true ;      ! SET TO MAP THE MODULE ON PASS 2
774 1263 2
775 1264 2
776 1265 2      begin
777 1266 2          bind modidstring = mhdrec [mhd$_name] +      ! POINT TO THE MODULE ID IN HEADER
778 1267 2              .mhdrec [mhd$b_namlng] : vector [, byte],      ! RECORD (COUNTED STRING)
779 1268 2              modcredat = modidstring [0] +      ! POINT TO THE MODULE CREATION DATE/TIME
780 1269 2                  .modidstring [0] + 1 : vector [, byte] ;
781 1270 2
782 1271 2      local bincredat : vector [2, long] ;
783 1272 2
784 1273 2      if .lnk$_curclu [clu$_shring]      ! IF THIS IS A SHAREABLE IMAGE THAT
785 1274 2          and      ! WAS PICKED UP FROM A
786 1275 2          .(lnk$_curclu [clu$_credat]) < 0, 32, 0 > neq 0      ! SHAREABLE IMAGE LIBRARY
787 1276 2      then begin      ! (CREDAT WILL BE 0 IF NOT)
788 1277 2          lnk$_bintim (modcredat, bincredat) ;      ! CONVERT TIME TO BINARY
```

```

: 789      1278  4      if not .(lnk$gl_curclu [clu$gsmatch])<31, 1, 0>      ! IF HIGH BIT CLEAR THEN DO DATE CHECK
: 790      1279  4      then if not ch$eql (8, bincredat, 8, lnk$gl_curclu [clu$q_credat])      ! IF THE DATES ARE DIFFERENT
: 791      1280  4      then signal (lin$ datmismch, 4, bincredat      ! THEN WARN THE USER,
: 792      1281  4      , lnk$gl_curfil [fdb$q_filename]      ! BUT CONTINUE
: 793      1282  4      , lnk$gl_curclu [clu$q_credat]
: 794      1283  4      , lnk$gl_curfil [fdb$q_libnamdsc]
: 795      1284  4      ;
: 796      1285  4      ch$fill (0, 8, lnk$gl_curclu [clu$q_credat]) ;      ! ZERO QUADWORD DATE/TIME, SINCE IT
: 797      1286  4      end ;      ! GETS REUSED
: 798      1287  3
: 799      1288  3
: 800      1289  3      if not .obmodesc [omd$sv_debugger]      ! IF THIS IS NOT A DEBUGGER MODULE
: 801      1290  3      and
: 802      1291  4      (.lnk$gt_imgid [0] eql 0 or .lnk$gl_tfrpsc eql 0)      ! IF THERE IS NO IMAGE IDENT SET UP
: 803      1292  4      then begin
: 804      1293  4      imageidstring [0] = minu (.modidstring [0]
: 805      1294  4      , sym$sc_maxlng) ;      ! EXTRACT THE LENGTH MINIMIZED WITH
: 806      1295  4      if .imageidstring [0] neq 0      ! MAXIMUM ALLOWED STRING LENGTH
: 807      1296  4      then ch$move (.imageidstring [0]      ! AND IF NOT NULL
: 808      1297  4      , modidstring [1]      ! SAVE IT TILL END OF MODULE SO THAT
: 809      1298  4      , imageidstring [1]      ! MAY USE IT IF THIS HAS A TRANSFER
: 810      1299  4      ;      ! ADDRESS
: 811      1300  3      end ;
: 812      1301  2      end ;
: 813      1302  2      return true ;
: 814      1303  1      end ;

```

! OF MODULE HEADER PROCESSING

		0FFC 00000	PROHDR:	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	1164
	5B	00000000G	00 9E 00002	MOVAB	LIB\$SIGNAL, R11	
	5A	00000000G	00 9E 00009	MOVAB	LNK\$GL_CURFIL, R10	
	59	00000000'	EF 9E 00010	MOVAB	OBMODE\$C+20, R9	
	5E		08 C2 00017	SUBL2	#8, SP	
	56	BC	A9 D0 0001A	MOVL	OBJREC, R6	1187
FF67	CF		00 FB 0001E	CALLS	#0, SEQCHK	1191
	56		50 E9 00023	BLBC	R0, 3\$	
	50	BC	A9 D0 00026	MOVL	OBJREC, R0	1193
		01	A0 95 0002A	TSTB	1(R0)	
			03 13 0002D	BEQL	1\$	
		0197	31 0002F	BRW	14\$	
		00000000'	EF B6 00032	INCW	LNK\$GW_NMODULES	1195
		02	A6 95 00038	TSTB	2(R6)	1197
			1A 13 0003B	BEQL	2\$	
7E	6A		14 C1 0003D	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	1201
		05	A6 9F 00041	PUSHAB	5(R6)	
	7E	02	7E D4 00044	CLRL	-(SP)	
			A6 9A 00046	MOVZBL	2(R6), -(SP)	
		04	DD 0004A	PUSHL	#4	
		00000000G	8F DD 0004C	PUSHL	#LIN\$ STRLVL	
	6B		06 FB 00052	CALLS	#6, LIB\$SIGNAL	
			77 11 00055	BRB	7\$	1203
	50	03	A6 3C 00057	MOVZWL	3(R6), R0	1206
B4	A9		50 B0 0005B	MOVW	R0, MAXRECLNG	
0800	8F		50 B1 0005F	CPW	R0, #2048	

7E	6A	05	18	1B	00064	BLEQU	4\$		1210
			14	C1	00066	ADDL3	#20, LNK\$GL_CURFIL, -(SP)		1209
	7E	B4	A6	9F	0006A	PUSHAB	5(R6)		1210
			A9	3C	0006D	MOVZWL	MAXRECLNG, -(SP)		
			03	DD	00071	PUSHL	#3		
	6B	00000000G	8F	DD	00073	PUSHL	#LINS_ILLRECLN		
			05	FB	00079	CALLS	#5, LIB\$SIGNAL		1212
	52	00000000G	50	11	0007C	BRB	7\$		1215
			8F	DD	0007E	MOVL	#LNK\$K_MAX_FILENAME_LENGTH, -		
							MODULE_MAX_LENGTH		
	51	05	A6	9A	00085	MOVZBL	5(R6), R1		1217
			26	13	00089	BEQL	6\$		
12	50	00000000G	00	DD	0008B	MOVL	LNK\$GL_CURCLU, R0		1219
	A0		02	E1	00092	BBC	#2, 88(R0), 5\$		
	8F		51	D1	00097	CMPL	R1, #LNK\$K_MAX_FILENAME_LENGTH		1220
			31	1B	0009E	BLEQU	8\$		
	52	00000000G	8F	DD	000A0	MOVL	#LNK\$K_MAX_FILENAME_LENGTH, -		1222
							MODULE_MAX_LENGTH		
	1F		08	11	000A7	BRB	6\$		
			51	91	000A9	CMPB	R1, #31		1226
	52		23	1B	000AC	BLEQU	8\$		
7E	6A		1F	DD	000AE	MOVL	#31, MODULE_MAX_LENGTH		1228
		05	14	C1	000B1	ADDL3	#20, LNK\$GL_CURFIL, -(SP)		1238
			A6	9F	000B5	PUSHAB	5(R6)		1237
			06	BB	000B8	PUSHR	#^M<R1,R2>		1238
		05	A6	9F	000BA	PUSHAB	5(R6)		1236
		00000000'	EF	9F	000BD	PUSHAB	LNK\$GT_MODSTRING		1235
			06	DD	000C3	PUSHL	#6		1238
	6B	00000000G	8F	DD	000C5	PUSHL	#LINS_ILLNAMELEN		
			08	FB	000CB	CALLS	#8, LIB\$SIGNAL		
			00FC	31	000CE	BRW	15\$		1240
	57		6A	DD	000D1	MOVL	LNK\$GL_CURFIL, R7		1243
	A9		57	DD	000D4	MOVL	R7, OBMODESC+4		
FO	A9	00000000G	00	DD	000D8	MOVL	LNK\$AL_RAB+16, OBMODESC+12		1244
F8	A9	00000000G	00	DD	000E0	MOVW	LNK\$AL_RAB+20, OBMODESC+16		1245
FC	58	05	A6	9A	000E8	MOVZBL	5(R6), R8		1246
	A9		58	90	000EC	MOVB	R8, OBMODESC+40		
1F	00		58	2C	000F0	MOVCS	R8, 6(R6), #0, #31, OBMODESC+41		1249
	06		A9	01	AE	MNEGW	#1, OBMODESC+18		1251
		15	01	AE	000FC	MNEGW	#1, OBMODESC+22		1252
	FE		8F	90	00100	MOVB	#-125, OBMODESC+20		1254
	02		02	EF	00104	EXTZV	#2, #1, 10(R7), R0		1255
50	OA	A7	01	50	FO	INSV	R0, #2, #1, OBMODESC+20		
69		01	03	EF	0010F	EXTZV	#3, #1, 10(R7), R0		1257
50	OA	A7	01	50	FO	INSV	R0, #3, #1, OBMODESC+20		
69		01	05	EF	0011A	EXTZV	#5, #1, 10(R7), R0		1258
50	OA	A7	01	50	FO	INSV	R0, #5, #1, OBMODESC+20		
69		01	A9	00000000'	EF	3C	00125		1259
		08	00	03	E0	0012D	MOVZWL	LNK\$GW_NMODULES, OBMODESC+28	
	03	00000000G	69	10	88	00135	BBS	#3, LNK\$GL_CTLMSK+1, 9\$	1261
			56	06	A846	9E	00138	9\$:	1262
			57	66	9A	0013D	MOVAB	6(R8)[R6], R6	1266
			50	00000000G	00	DD	00140	MOVZBL	(R6), R7
			A0	02	E1	00147	MOVL	LNK\$GL_CURCLU, R0	1269
4E	58						BBC	#2, 88(R0), 11\$	1273
			30	A0	D5	0014C	TSTL	48(R0)	
			49	13	0014F	BEQL	11\$		1275

				5E	DD	00151	PUSHL	SP		1277
		01	A746	9F	00153		PUSHAB	1(R7)[R6]		
	00000000G	00		02	FB	00157	CALLS	#2, LNK\$BINTIM		
		54	00000000G	00	D0	0015E	MOVL	LNK\$GL_CURCLU, R4		1279
			0087	C4	95	00165	TSTB	135(R4)		
				21	19	00169	BLSS	10\$		
30	A4	6E		08	29	0016B	CMPC3	#8, BINCREDAT, 48(R4)		1280
				1A	13	00170	BEQL	10\$		
		50		6A	D0	00172	MOVL	LNK\$GL_CURFIL, R0		1284
			1C	A0	9F	00175	PUSHAB	28(R0)		
			30	A4	9F	00178	PUSHAB	48(R4)		1283
			14	A0	9F	0017B	PUSHAB	20(R0)		1282
			0C	AE	9F	0017E	PUSHAB	BINCREDAT		1281
				04	DD	00181	PUSHL	#4		1284
			00000000G	8F	DD	00183	PUSHL	#LINS DATMISMCH		
		6B		06	FB	00189	CALLS	#6, LIB\$SIGNAL		
08		50	00000000G	00	D0	0018C	MOVL	LNK\$GL_CURCLU, R0		1286
	00	6E		00	2C	00193	MOVC5	#0, (SP), #0, #8, 48(R0)		
			30	A0		00198				
		69		05	E0	0019A	BBS	#5, OBMODESC+20, 14\$		1289
	2B		00000000G	00	95	0019E	TSTB	LNK\$GT_IMGID		1291
				08	13	001A4	BEQL	12\$		
			00000000'	EF	D5	001A6	TSTL	LNK\$GL_TFRPSC		
				1B	12	001AC	BNEQ	14\$		
		50		57	D0	001AE	MOVL	R7, R0		1293
		1F		50	91	001B1	CMPB	R0, #31		
				03	1B	001B4	BLEQU	13\$		
		50		1F	D0	001B6	MOVL	#31, R0		
		CC	A9	50	90	001B9	MOVB	R0, IMAGEIDSTRING		
			50	CC	A9	001BD	MOVZBL	IMAGEIDSTRING, R0		1295
				06	13	001C1	BEQL	14\$		
CD	A9	01	A6	50	28	001C3	MOVC3	R0, 1(R6), IMAGEIDSTRING+1		1298
			50	01	D0	001C9	MOVL	#1, R0		1302
					04	001CC	RET			
				50	D4	001CD	CLRL	R0		1303
				04	001CF		RET			

; Routine Size: 464 bytes, Routine Base: \$CODE\$ + 035C

; 815 1304 1


```

817 1305 1 routine delete_psect (psectname, retdescadr) =
818 1306     begin
819 1307
820 1308     THIS ROUTINE SEARCHES ALL CLUSTERS FOR ANOTHER DEFINITION OF THE
821 1309     NAMED GLOBAL PSECT. IF ONE IS FOUND, IT IS DELETED (BY SETTING
822 1310     A FLAG IN THE DESCRIPTOR). IF THE CLUSTER IT IS FOUND IN IS
823 1311     A SHAREABLE IMAGE, THEN AN ERROR MESSAGE IS ISSUED, SINCE A
824 1312     PSECT CANNOT BE CONTAINED IN TWO SHAREABLE IMAGES AT THE SAME
825 1313     TIME.
826 1314
827 1315     map
828 1316         psectname : ref vector [, byte],
829 1317         retdescadr : ref vector [, long];
830 1318     local
831 1319         cludesc : ref block [, byte],
832 1320         fdb : ref block [, byte],
833 1321         pdesc : ref block [, byte];
834 1322
835 1323     cludesc = lnk$gl_clulst;
836 1324
837 1325     LOOP OVER ALL CLUSTERS, SEARCHING THE GLOBAL PSECT LIST FOR
838 1326     A MATCH
839 1327
840 1328     while (cludesc = .cludesc [clu$l_nxtclu]) neq 0 do
841 1329
842 1330         if lib$lookup_tree (cludesc [clu$l_gpslst], .psectname, lnk$compare_pscnod, pdesc) and not (pdesc =
843 1331             .pdesc [node$l_ptr]; ! GET PSECT DESCRIPTOR ADDRESS
844 1332             .pdesc [psc$deleted]) ! TEST IF PSECT HAS BEEN DELETED
845 1333
846 1334         then
847 1335             begin
848 1336                 pdesc [psc$deleted] = true; ! FLAG PSECT DELETED FROM THIS CLUSTER
849 1337
850 1338                 if .cludesc [clu$v_shring] ! IF CLUSTER IS A SHAREABLE IMAGE
851 1339                 then
852 1340                     begin
853 1341                         fdb = .cludesc [clu$l_fstfdb]; ! GET FDB ADDRESS FOR SHAREABLE IMAGE
854 1342                         signal (lin$mulshrpsc, 3, .psectname, ! ISSUE ERROR MESSAGE
855 1343                             fdb [fdb$q_filename], lnk$gl_curfil [fdb$q_filename], lin$noimgfil);
856 1344                         lnk$gl_ctlmsk [lnk$v_image] = false; ! IMAGE WOULD ONLY BE INVALID
857 1345                     end;
858 1346
859 1347                 retdescadr [0] = .pdesc; ! RETURN ADDRESS TO CALLER
860 1348                 return true;
861 1349             end;
862 1350
863 1351     return false
864 1352 end;
```

```

                                001C 00000 DELETE_PSECT:
                                .WORD      Save R2,R3,R4           : 1305
5E                               SUBL2      #4, SP
52 00000000G 00 9E 00005      MOVAB      LNK$GL_CLULST, CLUDESC : 1323
```

52		62	D0	0000C	1\$:	MOVL	(CLUDESC), CLUDESC	:	1329
		6A	13	0000F		BEQL	3\$	:	
		5E	DD	00011		PUSHL	SP	:	1331
	00000000G	00	9F	00013		PUSHAB	LNK\$COMPARE_PSCNOD	:	
		04	AC	DD	00019	PUSHL	PSECTNAME	:	
		14	A2	9F	0001C	PUSHAB	20(CLUDESC)	:	
	00000000G	00	04	FB	0001F	CALLS	#4, LIB\$LOOKUP_TREE	:	
		E3	50	E9	00026	BLBC	R0, 1\$	:	
		50	6E	D0	00029	MOVL	PDESC, R0	:	1332
		6E	0A	A0	D0	MOVL	10(R0), PDESC	:	
		50	6E	D0	0002C	MOVL	PDESC, R0	:	1333
D4	0B	A0	06	E0	00033	BBS	#6, 11(R0), 1\$	:	
		53	6E	D0	00038	MOVL	PDESC, R3	:	1336
	0B	A3	8F	88	0003B	BISB2	#64, 11(R3)	:	
2E	58	A2	02	E1	00040	BBC	#2, 88(CLUDESC), 2\$	:	1338
		54	A2	D0	00045	MOVL	8(CLUDESC), FDB	:	1341
			8F	DD	00049	PUSHL	#LINS NOIMGFIL	:	1343
7E	00000000G	00	14	C1	0004F	ADDL3	#20, [LNK\$GL_CURFIL, -(SP)	:	
			14	A4	9F	PUSHAB	20(FDB)	:	
			04	AC	DD	PUSHL	PSECTNAME	:	
			03	DD	0005D	PUSHL	#3	:	
			8F	DD	0005F	PUSHL	#LINS MULSHRPSC	:	
	00000000G	00	06	FB	00065	CALLS	#6, LIB\$SIGNAL	:	
	00000000G	00	01	8A	0006C	BICB2	#1, LNK\$GL_CTLMSK	:	1344
	08	BC	53	D0	00073	MOVL	R3, @RETDESCADR	:	1347
		50	01	D0	00077	MOVL	#1, R0	:	1348
			04	0007A		RET		:	
			50	D4	0007B	CLRL	R0	:	1351
			04	0007D	3\$:	RET		:	1352

; Routine Size: 126 bytes, Routine Base: \$CODE\$ + 052C


```

: 866      1353 1 routine fndpscmapent (psctnum, omdptr) =
: 867      1354 2   begin
: 868      1355 2
: 869      1356 2       THIS ROUTINE RETURNS THE ADDRESS OF THE MAPPING
: 870      1357 2       TABLE ENTRY FOR P-SECTION NUMBER "PSCTNUM" IN
: 871      1358 2       THE CURRENT MODULE.
: 872      1359 2       THE P-SECT MAPPING TABLE IS AN ARRAY APPENDED
: 873      1360 2       TO THE MODULE DESCRIPTOR BLOCK. IT IS INITIALLY
: 874      1361 2       ALLOCATED WITH SPACE SUFFICIENT FOR 256
: 875      1362 2       ENTRIES. IF MORE THAN 256 PSECTS ARE ENCOUNTERED DURING
: 876      1363 2       THE PROCESSING OF THE OBJECT MODULE, THE INITIAL TABLE IS
: 877      1364 2       COPIED OUT TO ANOTHER BLOCK, AND THE MAP TABLE APPENDED TO
: 878      1365 2       THE OBJECT MODULE DESCRIPTOR BLOCK IS A TABLE OF TABLE ADDRESSES.
: 879      1366 2
: 880      1367 2       THE SECOND ARGUMENT IS OPTIONAL. IF NOT SUPPLIED, THE OWN OBJ MODULE
: 881      1368 2       DESCRIPTOR (OBMODESC) WILL BE USED.
: 882      1369 2
: 883      1370 2   literal
: 884      1371 2   tablesize = 256*pmt$c_size;           ! SIZE IN BYTES OF A BLOCK
: 885      1372 2   local
: 886      1373 2   blockoff;                             ! OFFSET OF ENTRY IN EXTENDED BLOCK
: 887      1374 2   first_entry : ref block[, byte],
: 888      1375 2   omdesc : ref block[, byte],             ! POINTER TO OBJ MODULE DESCRIPTOR
: 889      1376 2   mapent : ref block[, byte],             ! ADDRESS IN PSECT MAPPING TABLE
: 890      1377 2   first256 : ref block[, byte],           ! address of block of first 256
: 891      1378 2   mapoff : ref block[, byte];             ! ADDRESS OF EXTENDED MAPPING BLOCK
: 892      1379 2   builtin
: 893      1380 2   nullparameter;
: 894      1381 2   if nullparameter (2)                   ! SETUP POINTER TO OBJ MODULE DESCRIPTOR
: 895      1382 2   then
: 896      1383 2       omdesc = obmodesc
: 897      1384 2   else
: 898      1385 2       omdesc = .omdptr;
: 899      1386 2
: 900      1387 2   mapent = omdesc [omd$t_pscmap] + .psctnum*pmt$c_size;      ! GET ENTRY ADDRESS IN BASE MAPPING TABLE
: 901      1388 2
: 902      1389 2   if not .omdesc [omd$v_p256]              ! IF NOT INTO EXTENDED MAP TABLE
: 903      1390 2       and .psctnum lequ 255                ! AND LEQU 255 PSECTS
: 904      1391 2   then
: 905      1392 2       return .mapent
: 906      1393 2
: 907      1394 2   EXTENDED PSECT ENTRY
: 908      1395 2
: 909      1396 2   else
: 910      1397 2   begin
: 911      1398 2       blockoff = (.psctnum mod 256)*pmt$c_size;      ! OFFSET OF ENTRY WITHIN EXTENDED TABLE
: 912      1399 2       mapent = omdesc [omd$t_pscmap] + (.psctnum/256)*pmt$c_size;
: 913      1400 2       ! GET ADDRESS OF EXTENDED TABLE ADDRESS
: 914      1401 2
: 915      1402 2       if ..mapent neq 0                      ! IF EXTENDED TABLE PRESENT
: 916      1403 2       and .omdesc [omd$v_p256]             ! AND THERE REALLY IS AN EXTENDED TABLE
: 917      1404 2       then
: 918      1405 2         return ..mapent + .blockoff          ! THEN RETURN ENTRY ADDRESS
: 919      1406 2       else
: 920      1407 2       begin
: 921      1408 2         lnk$alloblk (tablesize, mapoff);      ! ALLOCATE EXTENDED TABLE BLOCK
: 922      1409 2
```

```

: 923      1410 4      if not .omdesc [omd$v_p256]      ! IF THIS IS THE FIRST ONE
: 924      1411 4      then
: 925      1412 5          begin
: 926      1413 5              first_entry = omdesc[omd$t_pscmap];
: 927      1414 5              lnk$alloblk(tablesize,first256);
: 928      1415 5              ch$move(tablesize,omdesc[omd$t_pscmap],.first256);      ! first, allocate space for first 25
: 929      1416 5              ch$fill (0, tablesize, omdesc [omd$t_pscmap]);      ! move data for first 256
: 930      1417 5              first_entry[pmt$l_secpmt] = .first256;      ! ZERO BASE TABLE
: 931      1418 5              omdesc [omd$v_p256] = true;      ! FLAG INTO EXTENDED TABLE
: 932      1419 4          end;
: 933      1420 4
: 934      1421 4      ch$fill (0, tablesize, .mapoff);      ! ZERO NEW BLOCK
: 935      1422 4
: 936      1423 4      !
: 937      1424 4      ! POINT ENTRY IN BASE TABLE TO NEW BLOCK
: 938      1425 4      !
: 939      1426 4      mapent [pmt$l_secpmt] = .mapoff;
: 940      1427 4      return .mapoff + .blockoff
: 941      1428 3      end;
: 942      1429 2
: 943      1430 1      end;
end;
```

```

                                07FC 00000 FNDPSCMAPENT:
                                .WORD
5A 00000000G 00 9E 00002      MOVAB      Save R2,R3,R4,R5,R6,R7,R8,R9,R10      : 1353
5E          08 C2 00009      SUBL2      LNK$ALLOBLK, R10
02          6C 91 0000C      CMPB      #8, SP
                                (AP), #2
                                1$
                                8(AP)
                                2$
57 00000000' EF 9E 00016 1$:      MOVAB      OBMODESC, OMDESC      : 1383
                                04 11 0001D      BRB      3$
57          08 AC D0 0001F 2$:      MOVL      OMDPTR, OMDESC      : 1385
50          04 AC D0 00023 3$:      MOVL      PSCTNUM, R0      : 1387
56          48 A740 7E 00027      MOVAQ     72(OMDESC)[R0], MAPENT
OD          00          06 E0 0002C      BBS      #6, 20(OMDESC), 4$      : 1389
000000FF 8F          50 D1 00031      Cmpl     R0, #255      : 1390
                                04 1A 00038      BGTRU     4$
50          56 D0 0003A      MOVL      MAPENT, R0      : 1397
                                04 0003D      RET
7E          00          01 7A 0003E 4$:      EMUL     #1, R0, #0, -(SP)      : 1398
51          51 8E 00000100 8F 7B 00043      EDIV     #256, (SP)+, R1, R1
59          51 51 03 78 0004C      ASHL     #3, R1, BLOCKOFF
50 00000100 8F C6 00050      DIVL2    #256, R0      : 1399
56          48 A740 7E 00057      MOVAQ     72(OMDESC)[R0], MAPENT
                                66 D5 0005C      TSTL     (MAPENT)      : 1402
                                0A 13 0005E      BEQL     5$
05          14 A7          06 E1 00060      BBC      #6, 20(OMDESC), 5$      : 1403
50          66          59 C1 00065      ADDL3    BLOCKOFF, (MAPENT), R0      : 1405
                                04 00069      RET      : 1407
                                5E DD 0006A 5$:      PUSHL     SP      : 1408
7E          0800 8F 3C 0006C      MOVZWL   #2048, -(SP)
6A          02 FB 00071      CALLS    #2, LNK$ALLOBLK
```


LNK_OBJPASS1
V04=000

E 14
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 34
(11)

0800	8F	04	BE	29	14	A7	06	E0	00074	BBS	#6, 20(OMDESC), 6\$	:	1410
			00			58	48	A7	9E 00079	MOVAB	72(OMDESC), FIRST_ENTRY	:	1413
							04	AE	9F 0007D	PUSHAB	FIRST256	:	1414
						7E	0800	8F	3C 00080	MOVZWL	#2048, -(SP)	:	
						6A		02	FB 00085	CALLS	#2, LNK\$ALLOBLK	:	
						A7	0800	8F	28 00088	MOVCS	#2048, 72(OMDESC), @FIRST256	:	1415
						6E		00	2C 00090	MOVCS	#0, (SP), #0, #2048, 72(OMDESC)	:	1416
							48	A7	00097			:	
						68	04	AE	D0 00099	MOVL	FIRST256, (FIRST_ENTRY)	:	1417
0800	8F				14	A7	40	8F	88 0009D	BISB2	#64, 20(OMDESC)	:	1418
			00			6E		00	2C 000A2	MOVCS	#0, (SP), #0, #2048, @MAPOFF	:	1421
							00	BE	000A9			:	
						66		6E	D0 000AB	MOVL	MAPOFF, (MAPENT)	:	1426
			50			6E		59	C1 000AE	ADDL3	BLOCKOFF, MAPOFF, R0	:	1427
								04	000B2	RET		:	1430

; Routine Size: 179 bytes, Routine Base: \$CODE\$ + 05AA

```

: 945      1431 1 routine alloc_omdnode (omdnum, retnodeadr, omdesc) =
: 946      1432 1
: 947      1433 1 THIS ROUTINE IS CALLED BY LIB$INSERT_TREE TO ALLOCATE
: 948      1434 1 STORAGE FOR A NODE
: 949      1435 1
: 950      1436 2 begin
: 951      1437 2 map
: 952      1438 2         retnodeadr      : ref vector [, long],
: 953      1439 2         omdesc       : ref block [, byte];
: 954      1440 2 local
: 955      1441 2         node         : ref block [, byte];
: 956      1442 2
: 957      1443 2 lnk$alloblk (node$long, node);
: 958      1444 2 node [node$l_ptr] = .omdesc;
: 959      1445 2 retnodeadr [0] = .node;
: 960      1446 2 return true
: 961      1447 1 end;

```

			0000 00000	ALLOC_OMDNODE:		
	5E	04	C2 00002	.WORD	Save nothing	: 1431
		5E	DD 00005	SUBL2	#4, SP	: 1443
		0E	DD 00007	PUSHL	SP	
00000000G	00	02	FB 00009	PUSHL	#14	
	50	6E	D0 00010	CALLS	#2, LNK\$ALLOBLK	
0A	A0	AC	D0 00013	MOVL	NODE, R0	: 1444
08	BC	50	D0 00018	MOVL	OMDESC, 10(R0)	: 1445
	50	01	D0 0001C	MOVL	R0, @RETNODEADR	: 1446
		04	0001F	RET	#1, R0	: 1447

; Routine Size: 32 bytes, Routine Base: \$CODE\$ + 065D

; 962 1448 1


```

: 964      1449 1 global routine lnk$compare_omd (omdnum, curnode, omdesc) =
: 965      1450 1
: 966      1451 1 : THIS ROUTINE IS CALLED BY LIB$INSERT TREE AND LIB$LOOKUP TREE
: 967      1452 1 : TO COMPARE CURRENT NODE WITH THE NUMBER WE ARE LOOKING FOR
: 968      1453 1
: 969      1454 2 begin
: 970      1455 2 map
: 971      1456 2     curnode : ref block [, byte];
: 972      1457 2 bind
: 973      1458 2     curomd = .curnode [node$l_ptr] : block [, byte];
: 974      1459 2
: 975      1460 2 if .omdnum lss .curomd [omd$l_omdnum]           : COMPARE THIS WITH CURRENT NODE
: 976      1461 2 then return -1                                : RETURN -1 IF LSS
: 977      1462 2 else if .omdnum egl .curomd [omd$l_omdnum]    : IF EQUAL
: 978      1463 2     then return 0                              : THEN RETURN 0
: 979      1464 2     else return 1                              : ELSE IT'S GTR, RETURN 1
: 980      1465 1 end;

```

			0000 00000	.ENTRY	LNK\$COMPARE_OMD, Save nothing	: 1449
	50	08	AC D0 00002	MOVL	CURNODE, R0	: 1458
	50	0A	A0 D0 00006	MOVL	10(R0), R0	
1C	A0	04	AC D1 0000A	CMPL	OMDNUM, 28(R0)	: 1460
			04 18 0000F	BGEQ	1\$	
	50		01 CE 00011	MNEGL	#1, R0	: 1462
			04 00014	RET		
			03 12 00015 1\$:	BNEQ	2\$	
			50 D4 00017	CLRL	R0	: 1463
			04 00019	RET		
	50		01 D0 0001A 2\$:	MOVL	#1, R0	: 1464
			04 0001D	RET		: 1465

; Routine Size: 30 bytes, Routine Base: \$CODE\$ + 067D

; 981 1466 1

```

: 983      1467 1 routine propsectdef =
: 984      1468 2 begin
: 985      1469 3 ++
: 986      1470 4     PROCESS P-SECTION DEFINITIONS AS FOLLOWS:
: 987      1471 5     (0) CHECK LEGAL P-SECTION NAME AND ALIGNMENT PARAMETER
: 988      1472 6     (1) SEARCH FOR P-SECTION DEFINED
: 989      1473 7     (2) IF DEFINED:
: 990      1474 8         1. CHECK COMPATIBLE ATTRIBUTES
: 991      1475 9         2. IF OVERLAYED, CHECK EQUAL ALIGNMENT
: 992      1476 10        3. MAXIMIZE P-SECTIONS'S BASE ALIGNMENT
: 993      1477 11        4. GO TO (3)
: 994      1478 12     IF NOT DEFINED:
: 995      1479 13         1. INSERT IN P-SECTION LIST
: 996      1480 14         2. COPY FLAGS AND ALIGNMENT
: 997      1481 15     (3) ALLOCATE A MODULE CONTRIBUTION ENTRY AND INSERT
: 998      1482 16         IT LEXICALLY IN THE LIST BY MODULE NAME
: 999      1483 17     (4) UPDATE CURRENT BASE BY THIS ALIGNMENT THEN COPY
1000      1484 18         CURRENT BASE INTO MAPPING TABLE ENTRY
1001      1485 19     (5) ADD THAT ROUND UP TO PSECTIONS ACCUMULATED LENGTH
1002      1486 20     (6) ADD P-SECTIONS LENGTH CONTRIBUTION TO THE ACCUMULATION
1003      1487 21         IF CONCATENATED, IF OVERLAYED, MAXIMIZE LENGTH.
1004      1488 22     (7) COPY CONTRIBUTION LENGTH INTO MAPPING TABLE ENTRY
1005      1489 23     (8) COPY ALIGNMENT INTO MAPPING TABLE
1006      1490 24     (9) COPY MODULE DESCRIPTOR ADDRESS INTO MAPPING TABLE
1007      1491 25     (10) COUNT THIS P-SECTION AND CLEAR NO P-SECTIONS FLAG
1008      1492 26     (11) IF NECESSARY EXTEND MODULE DESCRIPTOR FOR ANOTHER
1009      1493 27         P-SECTION AND INSERT ADDRESS OF P-SECTION
1010      1494 28         IN TABLE.
1011      1495 29     (12) IF ANY PREMATURE SYMBOL DEFINITIONS, REMOVE EACH FROM
1012      1496 30         LIST, ADDING P-SECTION BASE TO RELOCATABLE VALUES
1013      1497 31         AND LINKING THEM ON THE P-SECTION'S LIST OF SYMBOLS
1014      1498 32 --
1015      1499 33 local
1016      1500 34     psctdesc : ref block [, byte],           ! CURRENT P-SECT DESCRIPTOR
1017      1501 35     modpscontriblk : ref block [, byte],    ! MODULE CONTRIBUTION DATA BLOCK
1018      1502 36     lastmodpscontriblk : ref block [, byte],
1019      1503 37     psectname : ref vector [, byte],         ! POINTER TO PSECT NAME
1020      1504 38     newpsect,                                ! TRUE IF NEW PSECT
1021      1505 39     sgpstype,                                ! TRUE IF GSD$C SGPS
1022      1506 40     pscope,                                  ! P-SECTION SEARCH SCOPE
1023      1507 41     psflags,
1024      1508 42     prevdfound,                               ! TRUE IF PREV. DEF. OF COMMON FROM SHR IMAGE
1025      1509 43     delpscdesc : ref block [, byte],         ! POINTER TO DELETED PSECT DESCRIPTOR
1026      1510 44     psctmapent : ref block [, byte],         ! P-SECTION MAPPING TABLE ENTRY
1027      1511 45     around,                                  ! ROUND UP FOR ALIGNMENT
1028      1512 46     pddptr : ref block [, byte],
1029      1513 47     localpdd : block [pdd$c_size, byte],
1030      1514 48     premsymlst,                                ! SAVE PREMATURE SYMBOL LIST
1031      1515 49     premsym : ref block [, byte];            ! PREMATURELY DEFINED SYMBOL
1032      1516 50
1033      1517 51 bind
1034      1518 52     length = around,
1035      1519 53     psctdef = objvec [.gsdoffset] : block [, byte];
1036      1520 54
1037      1521 55 !
1038      1522 56 ! DETERMINE WHICH PSECT SUBRECORD TYPE THIS IS
1039      1523 57 !
```



```
1040 1524 2 sgpstype = false;
1041 1525 2
1042 1526 2 if .psctdef [gps$b_gsdtyp] eql gsd$sc_psc
1043 1527 2 then
1044 1528 2     psectname = psctdef [gps$b_namlng]
1045 1529 2 else
1046 1530 2     begin
1047 1531 2         psectname = psctdef [sgps$b_namlng];    ! SHAREABLE IMAGE PSECT
1048 1532 2         sgpstype = true;
1049 1533 2     end;
1050 1534 2
1051 1535 2
1052 1536 2     FIRST CHECK FOR LEGAL P-SECTION NAME
1053 1537 2
1054 1538 2
1055 1539 2 if .psectname [0] gtru sym$sc_maxlng    ! CHECK NAME WITHIN THE LEGAL
1056 1540 2 or .psectname [0] eql 0                ! RANGE FOR SYMBOL AND P-SECTION
1057 1541 2 then
1058 1542 2     begin
1059 1543 2         signal (lin$_illnamelen, 6,        ! NAMES AND IF NOT
1060 1544 2             lnk$gt_pscstring, .psectname, .psectname [0], sym$sc_maxlng,    ! ISSUE AN ERROR MESSAGE
1061 1545 2             obmode$sc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
1062 1546 2         return false;
1063 1547 2     end;
1064 1548 2
1065 1549 2
1066 1550 2 SEE IF PSECT DEFINED BY OPTION
1067 1551 2
1068 1552 2 ch$fill (0, pdd$sc_size, localpdd);    ! ZERO ALL FIELDS IN LOCAL PDD
1069 1553 2 localpdd [pdd$b_align] = %x'FF';        ! SET ITS ALIGNMENT TO ALL ONES
1070 1554 2 pddptr = localpdd;                    ! PRESET IN CASE NOT FOUND
1071 1555 2 psflags = .psctdef [gps$w_flags];        ! ASSUME NOT DEFINED BY OPTION
1072 1556 2
1073 1557 2 if lnk$srcpscdef (.psectname, pddptr)    ! SEE IF DEFINED BY OPTION
1074 1558 2 then
1075 1559 2     begin
1076 1560 2         pddptr [pdd$b_namlng] = .pddptr [pdd$b_namlng] or %x'80';    ! AND IF SO, MARK AS DEFINED
1077 1561 2         psflags = .pddptr [pdd$w_flags] or    ! SET THE FLAGS
1078 1562 2         (.psflags and not .pddptr [pdd$w_flgmsk]);    ! QUALIFIED BY THE MASK
1079 1563 2     end;
1080 1564 2
1081 1565 2
1082 1566 2 IF THIS IS A SHAREABLE IMAGE CLUSTER AND THIS IS A GLOBAL PSECT
1083 1567 2 (COMMON), THEN SEE IF DEFINED SOMEWHERE ELSE
1084 1568 2
1085 1569 2
1086 1570 2 if .lnk$gl_curclu [clu$v_shrimg] and (.psctdef [gps$w_flags] and psc$m_shrbits) eql psc$m_shrbits
1087 1571 2 then
1088 1572 2     prevdfound = delete_psect (.psectname, delpscdesc)
1089 1573 2 else
1090 1574 2     prevdfound = false;
1091 1575 2
1092 1576 2
1093 1577 2 CHECK ALIGNMENT
1094 1578 2
1095 1579 2
1096 1580 2 if .psctdef [gps$b_align] gtru obj$sc_pscalilim    ! IF THE ALIGNMENT IS GREATER
```

```
1097 1581 then
1098 1582 begin
1099 1583 signal (lin$pscali, 4, .psectname, .psctdef [gps$b_align], ! THAN THE MAXIMUM, ISSUE
1100 1584 obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]); ! ERROR AND QUIT
1101 1585 psctdef [gps$b_align] = obj$c_pscalilim; ! USE OBJ LANG MAX
1102 1586 end;
1103 1587
1104 1588
1105 1589 SEARCH FOR P-SECTION DEFINED, CREATING DESCRIPTOR IF NOT.
1106 1590
1107 1591
1108 1592 if .obmodesc [omd$v_shring] ! IF THIS IS A SHAREABLE IMAGE
1109 1593 then
1110 1594 pscope = 0 ! THEN P-SECTION SEARCH IS ONLY IN THIS CLUSTER
1111 1595 else
1112 1596 if (pscope = .psctdef [gps$v_gbl]) eql 0 ! ELSE DEPENDS ON PSECT SCOPE
1113 1597 then
1114 1598 pscope = .pddptr [pdd$w_flags] and gps$m_gbl; ! OR OPTION DEFINITION SCOPE
1115 1599
1116 1600 if lnk$findpscnam (.psectname, .psflags, .pscope, psctdesc) ! GET DESCRIPTOR ADDRESS
1117 1601 and .psctdesc [psc$v_usrpsc] ! AND PSECT WAS SEEN IN SOURCE
1118 1602 then
1119 1603 begin ! AND IF FOUND
1120 1604 newpsect = false;
1121 1605
1122 1606
1123 1607 ** THIS CHECK DISABLED UNTIL C ISSUE RESOLVED
1124 1608
1125 1609 if .psctdesc [psc$l_omdnum] eql .lnk$gw_nmodules ! IF PSECT ALREADY DEFINED IN THIS MODULE
1126 1610 then signal (lin$muldefpsc, 3, ! THEN ISSUE A WARNING MESSAGE
1127 1611 .psctdesc [psc$b_namlng]
1128 1612 .obmodesc [omd$b_namlng]
1129 1613 ,lnk$gl_curfil [fdb$q_filename]
1130 1614 );
1131 1615
1132 1616 ** END OF DISABLED CODE
1133 1617
1134 1618 psctdesc [psc$l_omdnum] = .lnk$gw_nmodules; ! SET DEFINED IN CURRENT MODULE
1135 1619
1136 1620 if not .psctdesc [psc$v_optpsc] ! IF PSECT NOT MODIFIED BY OPTION
1137 1621 then
1138 1622 begin
1139 1623 if .psctdef [gps$w_flags] neq ((.psctdesc [psc$w_flags] ! CHECK COMPATIBLE
1140 1624 and not (psc$m_supres or psc$m_usrpsc)) and psc$m_pscbits)
1141 1625 then
1142 1626 signal (lin$mulpsc, 3, ! ATTRIBUTES, ISSUING ERROR
1143 1627 .psectname, obmodesc [omd$b_namlng], ! IF NOT
1144 1628 lnk$gl_curfil [fdb$q_filename]);
1145 1629
1146 1630 if .psctdesc [psc$v_ovr] ! IF OVERLAYED
1147 1631 then
1148 1632 if .psctdef [gps$b_align] neq .psctdesc [psc$b_align] ! CHECK EQUAL
1149 1633 then
1150 1634 signal (lin$ovrali, 3, ! ALIGNMENT
1151 1635 .psectname, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
1152 1636
1153 1637
```



```
1154 1638 4
1155 1639
1156 1640
1157 1641
1158 1642
1159 1643
1160 1644
1161 1645
1162 1646
1163 1647
1164 1648
1165 1649
1166 1650
1167 1651
1168 1652
1169 1653
1170 1654
1171 1655
1172 1656
1173 1657
1174 1658
1175 1659
1176 1660
1177 1661
1178 1662
1179 1663
1180 1664
1181 1665
1182 1666
1183 1667
1184 1668
1185 1669
1186 1670
1187 1671
1188 1672
1189 1673
1190 1674
1191 1675
1192 1676
1193 1677
1194 1678
1195 1679
1196 1680
1197 1681
1198 1682
1199 1683
1200 1684
1201 1685
1202 1686
1203 1687
1204 1688
1205 1689
1206 1690
1207 1691
1208 1692
1209 1693
1210 1694

end;                                ! PSECT NOT MODIFIED BY OPTION
if .psctdef [gps$b_align] gtru .psctdesc [psc$b_align] ! MAXIMIZE
and .pddptr [pdd$b_align] eql %x'FF'
then
    psctdesc [psc$b_align] = .psctdef [gps$b_align];    ! THE ALIGNMENT
end
P-SECTION IS NOT YET DEFINED - SO DO THAT
else
begin                                ! AND INSERT P-SECTION
    lnk$gw_npsects = .lnk$gw_npsects + 1;    ! AFTER COUNTING IT
    newpsect = true;
    psctdesc [psc$b_align] = .pddptr [pdd$b_align]; ! SET THE ALIGNMENT FROM PDD
    if .psctdesc [psc$b_align] eql %x'FF'    ! BUT IF NONE SPECIFIED
    then
        psctdesc [psc$b_align] = .psctdef [gps$b_align];    ! THEN USE INPUT FILE
    psctdesc [psc$w_flags] = .psflags;        ! SET THE FLAGS
    psctdesc [psc$v_usrpsc] = true;           ! FLAG DEFINED BY USER SOURCE
    psctdesc [psc$v_shring] = .lnk$gl_curclu [clu$v_shring];
                                                ! PROPOGATE SHR IMG FLAG INTO PSECT DESCRIPTOR
    psctdesc [psc$v_newdef] = .sgpstype;      ! REMEMBER IF SGPS DEFINITION
    if .lnk$gl_ctlmsk [lnk$v_shr]             ! IF CREATING A SHAREABLE IMAGE
    and not .obmodesc [omd$v_shring]          ! AND THIS IS NOT A SHAREABLE IMAGE INPUT MODULE
    and (.psctdesc [psc$w_flags] and psc$m_shrbits)
    eql psc$m_shrbits                        ! (MUST BE RELOCATABLE, GLOBAL AND OVERLAID)
    then
    begin
        lnk$gl_pshrnum = .lnk$gl_pshrnum + 1;    ! THEN COUNT IT FOR LNKSYPMOUT
        if .lnk$gl_pshrnum eql lnk$sc_maxpsects + 1 ! IF THERE ARE TOO MANY
        then
            signal_stop (lin$excpsec, 2, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename],
                lin$format);
        end;
    if .lnk$gl_ctlmsk [lnk$v_intfil]          ! IF THIS IS THE INTERNAL (SYSLIB) FILE
    and .lnk$gl_ctlmsk [lnk$v_supsys]        ! AND SYSLIB SUPPRESSION IS ON
    then
        psctdesc [psc$v_supres] = true;        ! THEN SUPPRESS THIS P-SECTION
    if .obmodesc [omd$v_shring]              ! IF A SHAREABLE IMAGE P-SECTION
    then
        if .sgpstype
        then
            psctdesc [psc$l_base] = .psctdef [sgps$l_base]
        else
            psctdesc [psc$l_base] = .psctdef [gps$l_alloc];
```

```

1211      1695      2
1212      1696      2
1213      1697      2
1214      1698      2
1215      1699      2
1216      1700      2
1217      1701      2
1218      1702      2
1219      1703      2
1220      1704      2
1221      1705      2
1222      1706      2
1223      1707      2
1224      1708      2
1225      1709      2
1226      1710      2
1227      1711      2
1228      1712      2
1229      1713      2
1230      1714      2
1231      1715      2
1232      1716      2
1233      1717      2
1234      1718      2
1235      1719      2
1236      1720      2
1237      1721      2
1238      1722      2
1239      1723      2
1240      1724      2
1241      1725      2
1242      1726      2
1243      1727      2
1244      1728      2
1245      1729      2
1246      1730      2
1247      1731      2
1248      1732      2
1249      1733      2
1250      1734      2
1251      1735      2
1252      1736      2
1253      1737      2
1254      1738      2
1255      1739      2
1256      1740      2
1257      1741      2
1258      1742      2
1259      1743      2
1260      1744      2
1261      1745      2
1262      1746      2
1263      1747      2
1264      1748      2
1265      1749      2
1266      1750      2
1267      1751      2

end;

! ITS BASE IS GIVEN BY SIZE FIELD IF NORMAL PSECT DEF

IF THIS IS A SHAREABLE IMAGE PSECT THAT WAS FOUND ELSEWHERE,
WE HAVE TO RUN AROUND AND FIX UP THE POINTERS IN THE PSECT
MAPPING TABLES FOR THE PREVIOUS DEF. TO POINT TO THIS NEW
DEFINITION. ALSO, PUT THE OLD PSECT MAPPING LIST ONTO THE
NEW DESCRIPTOR

if .prevdfound
then
begin
modpscontriblk = .delpscdesc [psc$l_mpc1st];      ! GET MPC LIST POINTER
psctdesc [psc$l_mpc1st] = .modpscontriblk;      ! SET INTO NEW DESCRIPTOR
psctdesc [psc$l_lstmpc] = .delpscdesc [psc$l_lstmpc]; ! COPY END OF LIST POINTER

while .modpscontriblk neq 0 do
begin
psctmapent = fndpsmapent (.modpscontriblk [mpc$w_pscnum], ! FIND PSECT MAPPING ENTRY
.modpscontriblk [mpc$l_owndmd]);
psctmapent [pmt$l_pscdes] = .psctdesc;      ! SET POINTER TO NEW DESCRIPTOR
modpscontriblk = .modpscontriblk [mpc$l_nxtmpc];
end;
end;

NOW TO CREATE A BLOCK OF DATA DESCRIBING THIS
MODULE'S CONTRIBUTION TO THE P-SECTION, LINK THE
BLOCK TO END OF THE P-SECTION MODULAR CONTRIBUTION LIST
AND POINT MODULE DESCRIPTOR
MAPPING TABLE ENTRY AT THIS BLOCK.

lnk$ailoblk (mpc$size, modpscontriblk);      ! ALLOCATE THE BLOCK
ch$fill (0, mpc$size, .modpscontriblk);      ! (make sure it's clean)
modpscontriblk [mpc$l_owndmd] = .obmodesc [omd$l_nxtomd]; ! LINK INTO TEMP LIST
obmodesc [omd$l_nxtomd] = .modpscontriblk;      ! TO BE FIXED UP AT PROEOM TIME
lastmodpscontriblk = .psctdesc [psc$l_lstmpc]; ! GET POINTER TO LAST CONTRIB. BLOCK FOR PSE

if .lastmodpscontriblk neq 0      ! IF THERE IS A LAST
then
lastmodpscontriblk [mpc$l_nxtmpc] = .modpscontriblk; ! THEN LINK NEW BLOCK INTO LIST

psctdesc [psc$l_lstmpc] = .modpscontriblk;      ! AND MAKE IT THE NEW LAST.

if .obmodesc [omd$w_hipsct] eql lnk$sc_maxpsects ! IF TOO MANY PSECTS ALREADY
and not .obmodesc [omd$w_nopsct]
then
signal_stop (lin$excpse, 2,      ! BEEN SEEN FOR THIS MODULE
obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$bq_filename], lin$format); ! ISSUE AN ERROR MESSAGE AND GIVE UP

obmodesc [omd$w_nopsct] = false;      ! CLEAR NO PSECTS FLAG
```



```

1268      1752 2      obmodesc [omd$w_hipsct] = .obmodesc [omd$w_hipsct] + 1;      ! COUNT NEW P-SECTION
1269      1753 2      modpscontriblk [mpc$w_pscnum] = .obmodesc [omd$w_hipsct];      ! SAVE PSECT NUMBER IN MPC
1270      1754 2      psctmapent = fndpsmapent (.obmodesc [omd$w_hipsct]);      ! GET MAP TABLE ENTRY
1271      1755 2      premsymlst = .psctmapent [pmt$l_symlst];      ! SAVE PREMATURE SYMBOL LIST
1272      1756 2      psctmapent [pmt$l_pscdes] = .psctdesc;      ! POINT IT TO PSECTION DESCRIPTOR
1273      1757 2      psctmapent [pmt$l_modcon] = .modpscontriblk;      ! AND TO THE NEW CONTRIBUTION BLOCK.
1274      1758 2      modpscontriblk [mpc$b_align] = .psctdef [gps$b_align];      ! COPY ITS ALIGNMENT
1275      1759 2      modpscontriblk [mpc$l_offset] = 0;      ! ASSUME ABSOLUTE AND/OR OVERLAYED
1276      1760 2      ! AND THUS 0 OFFSET CONTRIBUTION
1277      1761 2
1278      1762 2      if .psctdesc [psc$v_rel]      ! THAT IS ALL FOR ABSOLUTE
1279      1763 2      then
1280      1764 2      begin      ! P-SECTIONS
1281      1765 2
1282      1766 2      if not .psctdesc [psc$v_ovr]      ! IF A CONCATENATED PSECT NOT FROM SHR IMAGE
1283      1767 2      and not .obmodesc [omd$v_shring]
1284      1768 3      then
1285      1769 4      begin
1286      1770 4      around = (1^ .psctdef [gps$b_align]) - 1;
1287      1771 5      modpscontriblk [mpc$l_offset] = (.psctdesc [psc$l_base] + .around)
1288      1772 5      ! ADD IN THE ROUNDING AND TRUNCATE
1289      1773 4      and not .around;
1290      1774 4      around = .modpscontriblk [mpc$l_offset] - .psctdesc [psc$l_base];      ! THEN COMPUTE AMOUNT ADDED
1291      1775 4      psctdesc [psc$l_base] = .psctdesc [psc$l_base] + .around +      ! UPDATE THE BASE FOR NEXT
1292      1776 4      .psctdef [gps$l_alloc];      ! CONTRIBUTOR.
1293      1777 4      psctdesc [psc$l_length] = .psctdesc [psc$l_base];      ! WHICH IS ALSO TOTAL LENGTH
1294      1778 4      end
1295      1779 3      else
1296      1780 4      begin      ! FOR OVERLAYED
1297      1781 4
1298      1782 4      local
1299      1783 4      omd : ref block [, byte],
1300      1784 4      fdb : ref block [, byte];
1301      1785 4
1302      1786 5      if (.sgpstype      ! IF SHAREABLE IMAGE PSECT TYPE
1303      1787 5      and .psctdesc [psc$v_ovr] and .prevdfound      ! AND IT WAS DEFINED BEFORE IN OBJ MODULE
1304      1788 5      and (.delpscdesc [psc$l_length] gtru .psctdef [sgps$l_alloc]))
1305      1789 5      ! TO BE LARGER THAN ONE IN SHR IMG
1306      1790 4      then
1307      1791 5      begin
1308      1792 5      lib$lookup_tree (lnk$gl_omdtree, .delpscdesc [psc$l_omdnum],      ! FIND DEFINING MODULE OMD
1309      1793 5      lnk$compare_omd, omd);
1310      1794 5      omd = .omd [node$l_ptr];      ! POINT TO OMD FROM NODE
1311      1795 5      fdb = .omd [omd$l_owndb];      ! GET POINTER TO FDB FOR MODULE
1312      1796 5      signal (lin$shrp$clng, 6, .psectname, .delpscdesc [psc$l_length], omd [omd$b_namlng],
1313      1797 5      fdb [fdb$q_filename], .psctdef [sgps$l_alloc], lnk$gl_curfil [fdb$q_filename],
1314      1798 5      lin$noimgfil);
1315      1799 5      lnk$gl_cflmsk [lnk$v_image] = false;      ! DISABLE IMAGE PRODUCTION
1316      1800 4      end;
1317      1801 4
1318      1802 5      if (.psctdesc [psc$v_shring] and .psctdesc [psc$v_newdef]      ! IF PSECT DEFINED IN SHAREABLE IMAG
1319      1803 5      and .psctdesc [psc$v_ovr] and not .newpsect      ! AND THIS IS NOT FIRST DEFINITION
1320      1804 5      and (.psctdef [gps$l_alloc] gtru .psctdesc [psc$l_length]))
1321      1805 5      ! AND THIS DEFINITION IS BIGGER THAN SHR IMAGE ONE
1322      1806 4      then
1323      1807 5      begin      ! THAT IS AN ERROR
1324      1808 5      fdb = .psctdesc [psc$l_cludsc];      ! GET CLUSTER DESCRIPTOR POINTER
```

```
1325      fdb = .fdb [clu$l_fstfdb];      ! GET POINTER TO FDB FOR SHAREABLE IMAGE
1326      signal (lin$shrpscng, 6, .psectname, .psctdef [gps$l_alloc], ! SIGNAL THE PROBLEM
1327      obmodesc [omd$b_namng], lnk$gl_curfil [fdb$q_filename], .psctdesc [psc$l_length],
1328      fdb [fdb$q_filename], lin$noimgfil);
1329      lnk$gl_ctlmsk [lnk$v_image] = false; ! TURN OFF THE IMAGE
1330      end;
1331
1332      if not .obmodesc [omd$v_shring]      ! IF NOT A SHAREABLE IMAGE
1333      then
1334      begin
1335          if .psctdef [gps$l_alloc] gtru .psctdesc [psc$l_length] ! P-SECTION MAXIMIZE
1336          then
1337              psctdesc [psc$l_length] = .psctdef [gps$l_alloc]; ! THE ALLOCATION
1338          end
1339      else
1340      begin
1341          if .sgpstype      ! IF SPECIAL SHAREABLE IMAGE PSECT
1342          then
1343              psctdesc [psc$l_length] = .psctdef [sgps$l_alloc]; ! THEN SET LENGTH OF PSECT
1344          end;
1345      end;
1346
1347      if not .obmodesc [omd$v_shring]
1348      then
1349      begin
1350          modpscontriblk [mpc$l_length] = .psctdef [gps$l_alloc]; ! SET MODULE'S CONTRIBUTION
1351          obmodesc [omd$l_alloc] = .obmodesc [omd$l_alloc] + ! FINALLY ACCUMULATE THE CONTRIBUTION
1352          .modpscontriblk [mpc$l_length];
1353      end;
1354
1355      end;
1356
1357      NOW GO DOWN THE LIST OF PREMATURE SYMBOLS
1358      (THOSE AWAITING DEFINITION OF THIS PSECTION)
1359      AND RELOCATE THEM (IF THEY ARE RELOCATABLE)
1360      AND THEN LINK THEM ON TO THE P-SECTION LIST
1361
1362      while (premsym = .premsymlst) neq 0 do      ! RELOCATABLE AND THERE
1363      begin      ! ARE SOME PREMATURE
1364          if .premsym [sym$v_rel]      ! FOR EACH THAT IS RELOCATABLE
1365          then
1366              premsym [sym$l_value] = .premsym [sym$l_value] + ! ADD IN THE BASE OFFSET
1367              .modpscontriblk [mpc$l_offset]; ! OF THE MODULE'S
1368          premsymlst = .premsym [sym$l_psclst]; ! TAKE IT OFF LIST
1369          premsym [sym$l_psclst] = .psctdesc [psc$l_symlst]; ! AND PUT ON THE
1370          psctdesc [psc$l_symlst] = .premsym; ! P-SECTION SYMBOL LIST
1371      end;
1372
1373      1380
1374      1381
```



```
1382 1866 2 |
1383 1867 2 | COMPUTE LENGTH OF THIS GSD SUBRECORD
1384 1868 2 |
1385 1869 2 |
1386 1870 2 |
1387 1871 2 | if .sgpstype
1388 1872 2 | then
1389 1873 2 |     length = psctdef [sgps$t_name] - psctdef [sgps$t_start] + .psectname [0]
1390 1874 2 | else
1391 1875 2 |     length = psctdef [gps$t_name] - psctdef [gps$t_start] + .psectname [0];
1392 1876 2 | gsdoffset = .gsdoffset + .length;          ! UPDATE GSD OFFSET INTO RECORD
1393 1877 2 | return true;                                ! OF P-SECTION PROCESSOR
1394 1878 1 | end;
```

```
OFFC 00000 PROPSECTDEF:
5E 00000000' 2C C2 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 : 1467
58 00000000' EF 3C 00005 SUBL2 #44, SP : 1519
58 00000000' EF C0 0000C MOVZWL GSDOFFSET, R8
7E D4 00013 ADDL2 OBJVEC, R8 : 1524
68 95 00015 CLRL SGPSTYPE : 1526
06 12 00017 TSTB (R8)
56 08 A8 9E 00019 BNEQ 1$ : 1528
07 11 0001D MOVAB 8(R8), PSECTNAME
56 0C A8 9E 0001F 1$: MOVAB 12(R8), PSECTNAME : 1531
6E 01 D0 00023 MOVL #1, SGPSTYPE : 1532
1F 66 91 00026 2$: CMPB (PSECTNAME), #31 : 1539
04 1A 00029 BGTRU 3$
66 95 0002B TSTB (PSECTNAME) : 1540
2D 12 0002D BNEQ 4$
7E 00000000G 00 00000000' 14 C1 0002F 3$: ADDL3 #20, LNK$GL_CURFIL, -(SP) : 1545
1F DD 0003D PUSHAB OBMODESC+40
7E 66 9A 0003F PUSHL #31
00000000' 56 DD 00042 MOVZBL (PSECTNAME), -(SP)
00000000G 00 00000000G 06 DD 0004A PUSHL PSECTNAME : 1543
08 FB 00052 PUSHAB LNK$GT_PSCSTRING : 1545
0474 31 00059 PUSHL #6
00 00 2C 0005C 4$: CALLS #LINS_ILLNAMELEN : 1546
00 AE 00061 BRW 38$ : 1552
2E AE 00063 MNEGB #1, LOCALPDD+14 : 1553
0C AE 00067 MOVAB LOCALPDD, PDDPTR : 1554
52 02 A8 3C 0006C MOVZWL 2(R8), PSFLAGS : 1555
0C AE 9F 00070 PUSHAB PDDPTR : 1557
56 DD 00073 PUSHL PSECTNAME
00000000G 00 02 FB 00075 CALLS #2, LNK$SRCPSCDEF
18 50 E9 0007C BLBC R0, 5$
50 0C AE D0 0007F MOVL PDDPTR, R0 : 1560
OF A0 8F 88 00083 BISB2 #128, 15(R0)
51 51 A0 3C 00088 MOVZWL 12(R0), R1 : 1562
51 52 51 CB 0008C BICL3 R1, PSFLAGS, R1
```


		52	0A	A0	3C	00090	MOVZWL	10(R0), PSFLAGS	
		52		51	C8	00094	BISL2	R1, PSFLAGS	
20	58	50	00000000G	00	D0	00097	5\$:	MOVL	LNK\$GL_CURCLU, R0
		50		02	E1	0009E	BBC	#2, 88(R0), 6\$	1570
		50		02	A8	3C	000A3	MOVZWL	2(R8), R0
		50	FFFFFFCB	8F	CA	000A7	BICL2	#-53, R0	
		34		50	D1	000AE	CMPL	R0, #52	
				10	12	000B1	BNEQ	6\$	
				10	AE	9F	000B3	PUSHAB	DELPSDESC
				56	DD	000B6	PUSHL	PSECTNAME	1572
	FDD4	CF		02	FB	000B8	CALLS	#2, DELETE PSECT	
	04	AE		50	D0	000BD	MOVL	R0, PREVDFOUND	
				03	11	000C1	BRB	7\$	
				04	AE	D4	000C3	6\$:	CLRL
		5B		01	A8	9E	000C6	7\$:	PREVDFOUND
		09		6B	91	000CA	MOVAB	1(R8), R11	1574
				25	1B	000CD	CMPB	(R11), #9	1580
7E	00000000G	00		14	C1	000CF	BLEQU	8\$	
		7E	00000000'	EF	9F	000D7	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	1584
				6B	9A	000DD	PUSHAB	OBMODESC+40	
				56	DD	000E0	MOVZBL	(R11), -(SP)	
				04	DD	000E2	PUSHL	PSECTNAME	
				8F	DD	000E4	PUSHL	#4	
	00000000G	00	00000000G	06	FB	000EA	PUSHL	#LIN\$ PSCALI	
		6B		09	90	000F1	CALLS	#6, LIB\$SIGNAL	
04	00000000'	EF		02	E1	000F4	MOVB	#9, (R11)	1585
				51	D4	000FC	BBC	#2, OBMODESC+20, 9\$	1592
				17	11	000FE	CLRL	PSCOPE	1594
51	02	A8		01	04	EF	00100	BRB	10\$
				50	0F	12	00106	EXTZV	#4, #1, 2(R8), PSCOPE
				51	AE	D0	00108	BNEQ	10\$
				51	0A	3C	0010C	MOVL	PDDPTR, R0
				51	8F	CA	00110	MOVZWL	10(R0), PSCOPE
				14	AE	9F	00117	BICL2	#-17, PSCOPE
					51	DD	0011A	10\$:	PUSHAB
					52	DD	0011C	PUSHL	PSCTDESC
					56	DD	0011E	PUSHL	PSCOPE
					04	FB	00120	PUSHL	PSFLAGS
	00000000G	00			50	E8	00127	PUSHL	PSECTNAME
		03			008B	31	0012A	CALLS	#4, LNK\$FNDPSCNAM
					AE	D0	0012D	BLBS	R0, 12\$
F4	0B	A0		03	E1	00131	11\$:	BRW	16\$
				08	AE	D4	00136	12\$:	MOVL
				53	AE	D0	00139	BBC	PSCTDESC, R0
				53	EF	3C	0013D	CLRL	#3, 11(R0), 11\$
	28	A3	00000000'	0A	A3	9E	00145	NEWPSECT	1605
		59		0A	A3	E0	00149	MOVZWL	PSCTDESC, R3
53		69		02	A8	3C	0014D	MOVZWL	LNK\$GW_MODULES, 40(R3)
		50			00	ED	00151	MOVAB	10(R3), R9
50		69			1F	13	00156	BBS	#10, (R9), 14\$
					14	C1	00158	MOVZWL	2(R8), R0
					EF	9F	00160	CMPZV	#0, #10, (R9), R0
7E	00000000G	00			56	DD	00166	BEQL	13\$
					03	DD	00168	ADDL3	#20, LNK\$GL_CURFIL, -(SP)
					8F	DD	0016A	PUSHAB	OBMODESC+40
					05	FB	00170	PUSHL	PSECTNAME
					02	E1	00177	PUSHL	#3
25	00000000G	00	00000000G					PUSHL	#LIN\$ MULPSC
		69						CALLS	#5, LIB\$SIGNAL
								BBC	#2, (R9), 14\$

1631

2C	A3	6B	91	0017B	CMPB	(R11), 44(R3)	1634
7E	00000000G	00	1F	13 0017F	BEQL	14\$	1637
		00000000'	14	C1 00181	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	
			EF	9F 00189	PUSHAB	OBMODESC+40	
			56	DD 0018F	PUSHL	PSECTNAME	
		00000000G	03	DD 00191	PUSHL	#3	
			8F	DD 00193	PUSHL	#LINS_OVRALI	
00000000G	00		05	FB 00199	CALLS	#5, LIB\$SIGNAL	
2C	A3		6B	91 001A0	CMPB	(R11), 44(R3)	1641
			0F	1B 001A4	BLEQU	15\$	
	50	0C	AE	D0 001A6	MOVL	PDDPTR, R0	1642
FF	8F	0E	A0	91 001AA	CMPB	14(R0), #255	
			04	12 001AF	BNEQ	15\$	
2C	A3		6B	90 001B1	MOVB	(R11), 44(R3)	1644
		00C4	31	001B5	BRW	21\$	1601
		00000000'	EF	B6 001B8	INCW	LNK\$GW NPSECTS	1652
08	AE		01	D0 001BE	MOVL	#1, NEWPSECT	1653
	53	14	AE	D0 001C2	MOVL	PSCTDESC, R3	1654
	50	0C	AE	D0 001C6	MOVL	PDDPTR, R0	
2C	A3	0E	A0	90 001CA	MOVB	14(R0), 44(R3)	
FF	8F	2C	A3	91 001CF	CMPB	44(R3), #255	1656
			04	12 001D4	BNEQ	17\$	
2C	A3		6B	90 001D6	MOVB	(R11), 44(R3)	1658
	59	0A	A3	9E 001DA	MOVAB	10(R3), R9	1660
	69		52	B0 001DE	MOVW	PSFLAGS, (R9)	
01	A9		08	88 001E1	BISB2	#8, 1(R9)	1661
	50	00000000G	00	D0 001E5	MOVL	LNK\$GL_CURCLU, R0	1662
51	01		02	EF 001EC	EXTZV	#2, #1-88(R0), R1	
69	01		51	F0 001F2	INSV	R1, #13, #1, (R9)	
	01		6E	F0 001F7	INSV	SGPSTYPE, #15, #1, (R9)	1664
4D	00000000G		02	E1 001FC	BBC	#2, LNK\$GL_CTLMSK, 18\$	1666
45	00000000'		02	E0 00204	BBS	#2, OBMODESC+20, 18\$	1667
			69	3C 0020C	MOVZWL	(R9), R0	1668
			8F	CA 0020F	BICL2	#-53, R0	
	50	FFFFFFCB	50	D1 00216	CMPL	R0, #52	1670
	34		36	12 00219	BNEQ	18\$	
		00000000G	00	D6 0021B	INCL	LNK\$GL_PSHRNUM	1673
00010000	8F	00000000G	00	D1 00221	CMPL	LNK\$GL_PSHRNUM, #65536	1675
			23	12 0022C	BNEQ	18\$	
		00000000G	8F	DD 0022E	PUSHL	#LINS_FORMAT	1677
7E	00000000G	00	14	C1 00234	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	
		00000000'	EF	9F 0023C	PUSHAB	OBMODESC+40	
			02	DD 00242	PUSHL	#2	
		00000000G	8F	DD 00244	PUSHL	#LINS_EXCPSC	
			05	FB 0024A	CALLS	#5, LIB\$STOP	
0C	00000000G	00	03	E1 00251	BBC	#3, LNK\$GL_CTLMSK+1, 19\$	1682
04	00000000G	00	06	E1 00259	BBC	#6, LNK\$GL_CTLMSK+1, 19\$	1683
	01	A9	10	88 00261	BISB2	#16, 1(R9)	1685
OF	00000000'	EF	02	E1 00265	BBC	#2, OBMODESC+20, 21\$	1687
		07	6E	E9 0026D	BLBC	SGPSTYPE, 20\$	1692
	18	A3	08	A8 D0 00270	MOVL	8(R8), 24(R3)	
			05	11 00275	BRB	21\$	
	18	A3	04	A8 D0 00277	MOVL	4(R8), 24(R3)	1694
	35	04	AE	E9 0027C	BLBC	PREVDFOUND, 23\$	1707
	50	10	AE	D0 00280	MOVL	DELPSCDESC, R0	1710
	18	AE	0C	A0 D0 00284	MOVL	12(R0), MODPSCONTRIBLK	
	53	14	AE	D0 00289	MOVL	PSCTDESC, R3	1711

13	00	00000000G	00	57	18	AE	DO	0028D	22\$:	MOV	MODPSCONTRIBLK, 12(R3)	1712
			57	18	AE	DO	00292	22\$:	MOV	16(R0), 16(R3)	1714	
			6E	18	AE	DO	00297	22\$:	MOV	MODPSCONTRIBLK, R2	1716	
				04	A2	DD	0029B	22\$:	BEQ	23\$	1717	
				11	A2	3C	0029D	22\$:	PUSHL	4(R2)	1716	
					02	FB	002A0	22\$:	MOVZWL	17(R2), -(SP)	1718	
					50	DO	002A4	22\$:	CALLS	#2, FNDPSCMAPENT	1719	
					53	DO	002A9	22\$:	MOV	R0, PSCTMAPENT	1714	
					62	DO	002AC	22\$:	MOV	R3, (PSCTMAPENT)	1732	
					E2	DO	002AF	22\$:	MOV	(R2), MODPSCONTRIBLK	1733	
					18	AE	9F	002B3	22\$:	BRB	22\$	1734
					13	DD	002B8	22\$:	PUSHAB	MODPSCONTRIBLK	1735	
					02	FB	002BA	22\$:	PUSHL	#19	1737	
					AE	DO	002C1	22\$:	CALLS	#2, LNK\$ALLOBLK	1739	
					00	2C	002C5	22\$:	MOV	MODPSCONTRIBLK, R7	1741	
					67		002CA	22\$:	MOV	#0, (SP), #0, #19, (R7)	1744	
					EF	DO	002CB	22\$:	MOV	OBMODESC, 4(R7)	1745	
					57	DO	002D3	22\$:	MOV	R7, OBMODESC	1748	
					AE	DO	002DA	22\$:	MOV	PSCTDESC, R3	1751	
					A3	DO	002DE	22\$:	MOV	16(R3), LASTMODPSCONTRIBLK	1752	
					03	13	002E2	22\$:	BEQ	24\$	1753	
					57	DO	002E4	22\$:	MOV	R7, (LASTMODPSCONTRIBLK)	1754	
					57	DO	002E7	22\$:	MOV	R7, 16(R3)	1755	
					EF	B1	002EB	22\$:	CMPW	OBMODESC+18, #65535	1756	
					2A	12	002F4	22\$:	BNEQ	25\$	1757	
					EF	E8	002F6	22\$:	BLBS	OBMODESC+20, 25\$	1758	
					8F	DD	002FD	22\$:	PUSHL	#LINS\$ FORMAT	1759	
					14	C1	00303	22\$:	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	1760	
					EF	9F	0030B	22\$:	PUSHAB	OBMODESC+40	1761	
					02	DD	00311	22\$:	PUSHL	#2	1762	
					8F	DD	00313	22\$:	PUSHL	#LINS\$ EXCPSC	1763	
					05	FB	00319	22\$:	CALLS	#5, LIB\$STOP	1764	
					01	8A	00320	22\$:	BICB2	#1, OBMODESC+20	1765	
					EF	B6	00327	22\$:	INCW	OBMODESC+18	1766	
					EF	B0	0032D	22\$:	MOVW	OBMODESC+18, 17(R7)	1767	
					EF	3C	00335	22\$:	MOVZWL	OBMODESC+18, -(SP)	1768	
					01	FB	0033C	22\$:	CALLS	#1, FNDPSCMAPENT	1769	
					50	DO	00341	22\$:	MOV	R0, PSCTMAPENT	1770	
					AA	DO	00344	22\$:	MOV	4(PSCTMAPENT), PREMSYMLST	1771	
					53	DO	00348	22\$:	MOV	R3, (PSCTMAPENT)	1772	
					57	DO	0034B	22\$:	MOV	R7, 4(PSCTMAPENT)	1773	
					6B	90	0034F	22\$:	MOVB	(R11), 16(R7)	1774	
					A7	D4	00353	22\$:	CLRL	8(R7)	1775	
					03	E0	00356	22\$:	BBS	#3, (R9), 26\$	1776	
					012B	31	0035A	22\$:	BRW	33\$	1777	
					02	E0	0035D	22\$:	BBS	#2, (R9), 27\$	1778	
					02	E0	00361	22\$:	BBS	#2, OBMODESC+20, 27\$	1779	
					6B	78	00369	22\$:	ASHL	(R11), #1, R2	1780	
					52	D7	0036D	22\$:	DECL	AROUND	1781	
					A3	C1	0036F	22\$:	ADDL3	24(R3), AROUND, R0	1782	
					52	CB	00374	22\$:	BICL3	AROUND, R0, 8(R7)	1783	
					A3	C3	00379	22\$:	SUBL3	24(R3), 8(R7), AROUND	1784	
					A3	C1	0037F	22\$:	ADDL3	24(R3), AROUND, R0	1785	
					B840	9E	00384	22\$:	MOVAB	24(R8)[R0], 24(R3)	1786	
					A3	DO	0038A	22\$:	MOV	24(R3), 28(R3)	1787	
					00E1	31	0038F	22\$:	BRW	32\$	1788	

6D	71	6E	E9	00392	27\$:	BLBC	SGPSTYPE, 28\$	1786
	69	02	E1	00395		BBC	#2, (R9), 28\$	1787
	69	AE	E9	00399		BLBC	PREVDFOUND, 28\$	1788
	50	AE	D0	0039D		MOVL	DELPSCDESC, R0	
04	A8	A0	D1	003A1		CMPL	28(R0), 4(R8)	
		5E	1B	003A6		BLEQU	28\$	
		AE	9F	003A8		PUSHAB	OMD	1792
		CF	9F	003AB		PUSHAB	LNK\$COMPARE_OMD	
	55	AE	D0	003AF		MOVL	DELPSCDESC, R5	
		A5	DD	003B3		PUSHL	40(R5)	
		EF	9F	003B6		PUSHAB	LNK\$GL_OMDTREE	
00000000G	00	04	FB	003BC		CALLS	#4, LIB\$LOOKUP_TREE	
	50	AE	D0	003C3		MOVL	OMD, R0	1794
1C	AE	A0	D0	003C7		MOVL	10(R0), OMD	
	50	AE	D0	003CC		MOVL	OMD, R0	1795
	5A	A0	D0	003D0		MOVL	4(R0), FDB	
7E 00000000G	00	8F	DD	003D4		PUSHL	#LINS NOIMGFIL	1797
		14	C1	003DA		ADDL3	#20, LNK\$GL_CURFIL, -(SP)	
		A8	DD	003E2		PUSHL	4(R8)	
		AA	9F	003E5		PUSHAB	20(FDB)	
		A0	9F	003E8		PUSHAB	40(R0)	1796
		A5	DD	003EB		PUSHL	28(R5)	1797
		56	DD	003EE		PUSHL	PSECTNAME	
		06	DD	003F0		PUSHL	#6	
		8F	DD	003F2		PUSHL	#LINS SHRPSCLNG	
00000000G	00	09	FB	003F8		CALLS	#9, LIB\$SIGNAL	
00000000G	00	01	8A	003FF		BICB2	#1, LNK\$GL_CTLMSK	1799
50	69	0D	E1	00406	28\$:	BBC	#13, (R9), -29\$	1802
		69	B5	0040A		TSTW	(R9)	
		4C	18	0040C		BGEQ	29\$	
48	69	02	E1	0040E		BBC	#2, (R9), 29\$	1803
	44	AE	E8	00412		BLBS	NEWPSECT, 29\$	
1C	A3	A8	D1	00416		CMPL	4(R8), 28(R3)	1804
		3D	1B	0041B		BLEQU	29\$	
	5A	A3	D0	0041D		MOVL	36(R3), FDB	1808
	5A	AA	D0	00421		MOVL	8(FDB), FDB	1809
		8F	DD	00425		PUSHL	#LINS NOIMGFIL	1812
		AA	9F	0042B		PUSHAB	20(FDB)	
		A3	DD	0042E		PUSHL	28(R3)	
7E 00000000G	00	14	C1	00431		ADDL3	#20, LNK\$GL_CURFIL, -(SP)	1811
		EF	9F	00439		PUSHAB	OBMODESC+40	
		A8	DD	0043F		PUSHL	4(R8)	1812
		56	DD	00442		PUSHL	PSECTNAME	
		06	DD	00444		PUSHL	#6	
		8F	DD	00446		PUSHL	#LINS SHRPSCLNG	
00000000G	00	09	FB	0044C		CALLS	#9, LIB\$SIGNAL	
00000000G	00	01	8A	00453		BICB2	#1, LNK\$GL_CTLMSK	1813
09 00000000'	EF	02	E0	0045A	29\$:	BBS	#2, OBMODESC+20, 30\$	1816
1C	A3	A8	D1	00462		CMPL	4(R8), 28(R3)	1820
		0A	1B	00467		BLEQU	32\$	
		03	11	00469		BRB	31\$	1822
	05	6E	E9	0046B	30\$:	BLBC	SGPSTYPE, 32\$	1828
	A3	A8	D0	0046E	31\$:	MOVL	4(R8), 28(R3)	1830
0D 00000000'	EF	02	E0	00473	32\$:	BBS	#2, OBMODESC+20, 33\$	1836
	OC	A8	D0	0047B		MOVL	4(R8), 12(R7)	1839
00000000'	EF	A7	C0	00480		ADDL2	12(R7), OBMODESC+8	1841
	50	54	D0	00488	33\$:	MOVL	PREMSYMLST, PREMSYM	1853

LNK_OBJPASS1
V04=000

G 15
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 49
(14)

04	0A	A0	18	13	0048B	BEQL	35\$	...	1856
		60	03	E1	0048D	BBC	#3, 10(PREMSYM), 34\$	...	1859
		54	08	A7	C0 00492	ADDL2	8(R7), (PREMSYM)	...	1861
	04	A0	04	A0	D0 00496	MOVL	4(PREMSYM), PREMSYMLST	...	1862
	14	A3	14	A3	D0 0049A	MOVL	20(R3), 4(PREMSYM)	...	1863
				50	D0 0049F	MOVL	PREMSYM, 20(R3)	...	1853
				E3	11 004A3	BRB	33\$	...	1870
50		10	6E	E9	004A5	BLBC	SGPSTYPE, 36\$	...	1872
		58	58	C3	004A8	SUBL3	R8, R8, R0	...	
		51	66	9A	004AC	MOVZBL	(PSECTNAME), R1	...	
		50	51	C0	004AF	ADDL2	R1, R0	...	
		52	0D	A0	9E 004B2	MOVAB	13(R0), LENGTH	...	
			0D	11	004B6	BRB	37\$	...	
		58	58	C2	004B8	SUBL2	R8, R8	...	1874
		50	66	9A	004BB	MOVZBL	(PSECTNAME), R0	...	
		58	50	C0	004BE	ADDL2	R0, R8	...	
00000000'		52	09	A8	9E 004C1	MOVAB	9(R8), LENGTH	...	
		EF	52	A0	004C5	ADDW2	LENGTH, GSDOFFSET	...	1876
		50	01	D0	004CC	MOVL	#1, R0	...	1877
				04	004CF	RET		...	
			50	D4	004D0	CLRL	R0	...	1878
				04	004D2	RET		...	

; Routine Size: 1235 bytes, Routine Base: \$CODE\$ + 069B


```
1396 1879 1 routine crefilter (definition, weakflag, altomd) =
1397 1880 2   begin
1398 1881 3
1399 1882 4     THIS ROUTINE FILTERS FROM THE CROSS REFERENCE THE SYMBOLS
1400 1883 5     OF THE DEBUGGER AND ALSO MASKS THEM FROM APPEARING IN ANY
1401 1884 6     OTHER SYMBOL TABLE OUTPUT. (STB FILE OR GST OF IMAGE)
1402 1885 7
1403 1886 8     IT ALSO DECIDES IF ANY CROSS REFERENCE IS TO BE DONE
1404 1887 9     AT ALL.
1405 1888 10
1406 1889 11     THE ARGUMENT "DEFINITION" HAS VALUE TRUE OR FALSE.
1407 1890 12
1408 1891 13   map
1409 1892 14     altomd : ref block [, byte];
1410 1893 15   builtin
1411 1894 16     nullparameter;
1412 1895 17   bind
1413 1896 18     objmodesc = (if nullparameter (3) then objmodesc ! IF NOT SPECIFIED, THEN USE THE LOCAL ONE
1414 1897 19                     else .altomd) : block [, byte]; ! IF SPECIFIED, USE IT
1415 1898 20
1416 1899 21   if .symtabent [sym$v_lclsym] ! NO CROSS REF OF MODULE LOCAL SYMBOLS
1417 1900 22   then
1418 1901 23     return true;
1419 1902 24
1420 1903 25   if .lnk$gl_curfil [fdb$v_debugger] ! IF THIS FILE CONTAINS THE DEBUGGER
1421 1904 26   then
1422 1905 27     begin
1423 1906 28       if .definition ! AND IF THIS IS A DEFINITION
1424 1907 29       then
1425 1908 30         symtabent [sym$v_supres] = true; ! SUPPRESS IT FROM STB OUTPUT
1426 1909 31
1427 1910 32       if .lnk$gl_ctlmsk [lnk$v_supdbg] ! IF ALSO SUPRESSING
1428 1911 33       then
1429 1912 34         return true; ! DEBUGGER SYMBOLS FROM MAP
1430 1913 35
1431 1914 36       end; ! THAT IS ALL
1432 1915 37
1433 1916 38   if .lnk$gl_ctlmsk [lnk$v_brief] ! IF A BRIEF MAP THEN
1434 1917 39   then
1435 1918 40     return true; ! ALSO ALL DONE
1436 1919 41
1437 1920 42   if .lnk$gl_ctlmsk [lnk$v_intfil] ! IF THIS IS THE INTERNAL (SYSLIB) FILE
1438 1921 43   and .lnk$gl_ctlmsk [lnk$v_supsys] ! AND ITS CONTENTS ARE SUPPRESSED
1439 1922 44   then
1440 1923 45
1441 1924 46     if not .lnk$gl_ctlmsk [lnk$v_cros] ! IF NO CROSS REFERENCE
1442 1925 47     or not .definition ! OR THIS IS A REFERENCE
1443 1926 48     or .symtabent [sym$v_supres] ! OR THE SYMBOL HAS BEEN SUPPRESSED
1444 1927 49     then
1445 1928 50       return true ! THEN ALL DONE
1446 1929 51     else
1447 1930 52       objmodesc [omd$v_mapmod] = true; ! BUT IF SYMBOL IS ENTERED MAKE SURE MODULE GETS MAPPED
1448 1931 53
1449 1932 54   if (.definition or .lnk$gl_ctlmsk [lnk$v_cros]) ! IF WE ARE GOING TO CREF THIS SYMBOL
1450 1933 55   then
1451 1934 56     begin
1452 1935 57
```

```
1453 1936 3
1454 1937
1455 1938
1456 1939
1457 1940
1458 1941
1459 1942
1460 1943
1461 1944
1462 1945
1463 1946
1464 1947
1465 1948
1466 1949
1467 1950
1468 1951
1469 1952
1470 1953
1471 1954
1472 1955
1473 1956
1474 1957
1475 1958
1476 1959
1477 1960
1478 1961
1479 1962
1480 1963
1481 1964
1482 1965
1483 1966 1

if .symntnam [snb$b_namlng] gtr .lnk$gl_maxsymsz
! IF THIS SYMBOL NAME IS BIGGER THAN BIGGEST SEEN
then
    lnk$gl_maxsymsz = .symntnam [snb$b_namlng]; ! THEN MAKE THIS THE BIGGEST
if .objmodesc [omd$b_namlng] gtr .lnk$gl_maxmodsz ! ALSO CHECK THE MODULE NAME LENGTH
then
    lnk$gl_maxmodsz = .objmodesc [omd$b_namlng];
end;

if .definition ! IF THIS IS A DEFINITION MUST
then
    begin
        crf$insrtkey (lnk$al_sytblfmt, symntnam [snb$b_namlng], ! AT LEAST ENTER THE VALUE
            symtabent [sym$l_value], .symtabent [sym$w_flags]);
        ! BY ADDRESS AND PASSING THE FLAGS AS VALUE FLAGS
        symtabent [sym$v_crosref] = true; ! FLAG SYMBOL HAS BEEN CROSS REFERENCED IF DEFINITION
    end;

if .lnk$gl_ctlmsk [lnk$v_cros] ! IF A CROSS REFERENCE IS BEING PRODUCED
then
    begin
        crf$insrtref (lnk$al_sytblfmt, symntnam [snb$b_namlng], ! THEN INSERT THIS REFERENCE
            objmodesc [omd$b_namlng], .weakflag, .definition);
        lnk$gw_ncrosrfs = .lnk$gw_ncrosrfs + 1; ! AND COUNT IT
    end;

return true
end;
```

00FC 00000 CREFILTER:

57	00000000G	00	9E	00002	WORD	Save R2,R3,R4,R5,R6,R7	1879
56	00000000G	00	9E	00009	MOVAB	LNK\$AL_SYTBLFMT, R7	
55	00000000G	00	9E	00010	MOVAB	LNK\$GL_MAXMODSZ, R6	
54	00000000G	EF	9E	00017	MOVAB	LNK\$GL_MAXSYMSZ, R5	
53	00000000G	00	9E	0001E	MOVAB	SYMENTNAM, R4	
03		6C	91	00025	MOVAB	LNK\$GL_CTLMSK, R3	
		05	1F	00028	CMPB	(AP), #3	1896
	0C	AC	D5	0002A	BLSSU	1\$	
		09	12	0002D	TSTL	12(AP)	
50	24	A4	9E	0002F	BNEQ	2\$	
52		50	D0	00033	MOVAB	OBMODESC, R0	
		04	11	00036	MOVL	R0, R2	
52	0C	AC	D0	00038	BRB	3\$	1897
51	FC	A4	D0	0003C	MOVL	ALOMD, R2	1899
03	0B	A1	E9	00040	MOVL	SYMTABENT, R1	
		009B	31	00044	BLBC	11(R1), 5\$	
50	00000000G	00	D0	00047	BRW	13\$	1903
0A	A0	05	E1	0004E	MOVL	LNK\$GL_CURFIL, R0	
04	04	AC	E9	00053	BBC	#5, 10(R0), 7\$	
					BLBC	DEFINITION, 6\$	1907

		0B	A1		01	20	88	00057		BISB2	#32, 11(R1)	:	1909
						A3	95	0005B	6\$:	TSTB	LNK\$GL_CTLMSK+1	:	1911
						E4	19	0005E		BLSS	4\$	:	
	7D	01	A3			01	E0	00060	7\$:	BBS	#1, LNK\$GL_CTLMSK+1, 13\$	:	1917
	16	01	A3			03	E1	00065		BBC	#3, LNK\$GL_CTLMSK+1, 8\$	:	1921
	11	01	A3			06	E1	0006A		BBC	#6, LNK\$GL_CTLMSK+1, 8\$	:	1922
						63	95	0006F		TSTB	LNK\$GL_CTLMSK	:	1925
						6F	18	00071		BGEQ	13\$	:	
			6B		04	AC	E9	00073		BLBC	DEFINITION, 13\$	:	1926
	66	0B	A1			05	E0	00077		BBS	#5, 11(R1), 13\$	:	1927
		14	A2			10	88	0007C		BISB2	#16, 20(R2)	:	1931
			04		04	AC	E8	00080	8\$:	BLBS	DEFINITION, 9\$	:	1933
						63	95	00084		TSTB	LNK\$GL_CTLMSK	:	
						1B	18	00086		BGEQ	11\$	:	
65			50			64	D0	00088	9\$:	MOVL	SYMENTNAM, R0	:	1937
	04	A0	08			00	ED	0008B		CMPZV	#0, #8, 4(R0), LNK\$GL_MAXSYMSZ	:	
						04	15	00091		BLEQ	10\$	:	
			65		04	A0	9A	00093		MOVZBL	4(R0), LNK\$GL_MAXSYMSZ	:	1940
66			08			00	ED	00097	10\$:	CMPZV	#0, #8, 40(R2), LNK\$GL_MAXMODSZ	:	1942
	28	A2				04	15	0009D		BLEQ	11\$	:	
			66		28	A2	9A	0009F		MOVZBL	40(R2), LNK\$GL_MAXMODSZ	:	1944
			1B		04	AC	E9	000A3	11\$:	BLBC	DEFINITION, 12\$	:	1948
			7E		0A	A1	3C	000A7		MOVZWL	10(R1), -(SP)	:	1952
						51	DD	000AB		PUSHL	R1	:	
	7E		64			04	C1	000AD		ADDL3	#4, SYMENTNAM, -(SP)	:	1951
						57	DD	000B1		PUSHL	R7	:	
		00000000G	00			04	FB	000B3		CALLS	#4, CRF\$INSRTKEY	:	1952
			50		FC	A4	D0	000BA		MOVL	SYMTABENT, R0	:	1954
		0C	A0			02	88	000BE		BISB2	#2, 12(R0)	:	
						63	95	000C2	12\$:	TSTB	LNK\$GL_CTLMSK	:	1957
						1C	18	000C4		BGEQ	13\$	:	
					04	AC	DD	000C6		PUSHL	DEFINITION	:	1961
					08	AC	DD	000C9		PUSHL	WEAKFLAG	:	
					28	A2	9F	000CC		PUSHAB	40(R2)	:	
	7E		64			04	C1	000CF		ADDL3	#4, SYMENTNAM, -(SP)	:	1960
						57	DD	000D3		PUSHL	R7	:	
		00000000G	00			05	FB	000D5		CALLS	#5, CRF\$INSRTREF	:	1961
					00000000'	EF	B6	000DC		INCW	LNK\$GW_NCROSRFS	:	1962
			50			01	D0	000E2	13\$:	MOVL	#1, R0	:	1965
						04	00	000E5		RET		:	1966

; Routine Size: 230 bytes, Routine Base: \$CODE\$ + 0B6E


```
1485 1967 1 routine prosymbol (entmskflg) =
1486 1968 1 ++
1487 1969 1 This routine does all the work of processing symbols on PASS 1, including
1488 1970 1 entry points, simple symbol definitions and references and procedure names.
1489 1971 1 the following operations are performed:
1490 1972 1
1491 1973 1 1. If the current object module is a selective search module, then only
1492 1974 1 definitions are considered. these are ignored unless there is a current
1493 1975 1 outstanding reference to that symbol.
1494 1976 1
1495 1977 1 2. If a definition already exists in the table and this incoming symbol is
1496 1978 1 a definition, produce an error unless they are equal, absolute symbols
1497 1979 1 or unless this is a selective search module.
1498 1980 1
1499 1981 1 3. If there is no symbol in the table, insert this one.
1500 1982 1
1501 1983 1 4. If the entry found is a reference and the in coming symbol is a
1502 1984 1 definition, the flags, data type and value are copied into the entry
1503 1985 1 after removing it from the undefined list.
1504 1986 1
1505 1987 1 5. If the p-section specification has been seen, the base of this module's
1506 1988 1 contribution to the p-section is added, and the symbol linked on to the
1507 1989 1 p-section symbol list.
1508 1990 1
1509 1991 1 6. If p-section has not been seen, find the mapping table entry and link
1510 1992 1 this symbol on to its list of prematurely defined symbols. The symbol
1511 1993 1 will be relocated when p-section is defined.
1512 1994 1
1513 1995 1 7. Undefined symbols are linked on to the undefined symbol list and a count
1514 1996 1 of them is kept if the references to them are not weak.
1515 1997 1
1516 1998 1 ENTMSKFLG = 1 if the symbol contains an entry mask,
1517 1999 1 0 otherwise.
1518 2000 1
1519 2001 1 --
1520 2002 1
1521 2003 1 --
1522 2004 1
1523 2005 2 begin
1524 2006 2 routine shringsym =
1525 2007 2 begin
1526 2008 2
1527 2009 2 THIS ROUTINE LINKS A SYMBOL TABLE ENTRY INTO THE LIST OF REFERENCED SYMBOLS
1528 2010 2 IN A SHAREABLE IMAGE. THE SYMBOL IS COUNTED ALSO.
1529 2011 2
1530 2012 2 bind shrimgclu = .symtabent [sym$l_cludsc] : block [, byte];
1531 2013 2
1532 2014 2 if .shrimgclu [clu$v_based] or .lnk$gl_defclu [clu$v_based]
1533 2015 2 or not .symtabent [sym$v_rel]
1534 2016 2 then return true;
1535 2017 2
1536 2018 2 if .shrimgclu [clu$l_shrsyms] eql 0 ! IF NO SYMBOLS YET THIS CLUSTER
1537 2019 2 then lnk$gl_shrings = .lnk$gl_shrings + 1; ! THEN COUNT THIS SHAREABLE IMAGE
1538 2020 2
1539 2021 2 shrimgclu [clu$l_shrsyms] = .shrimgclu [clu$l_shrsyms] + 1; ! COUNT THIS SYMBOL IN CLUSTER
1540 2022 2 lnk$gl_shrsyms = .lnk$gl_shrsyms + 1; ! COUNT IN TOTAL NUMBER OF SYMBOLS
1541 2023 2 symtabent [sym$l_shrlnk] = .shrimgclu [clu$l_shrlst]; ! LINK SYMBOL INTO LIST OFF CLUSTER
```



```
: 1542      2024  3      symtabent [sym$u_gref] = true;                ! FLAG SYMBOL HAS BEEN ENTERED INTO
: 1543      2025  3      shrinkclu [clu$l_shrlst] = .symtabent;
: 1544      2026  3      return true
: 1545      2027  2      end;
```

```
                                0000 00000 SHRIMGSYM:
                                .WORD
                                Save nothing
                                SYMTABENT, R1
                                32(R1), R0
                                88(R0), 2$
                                LNK$GL_DEFCLU+88, 2$
                                #3, 10(R1), 2$
                                44(R0)
                                1$
                                LNK$GL_SHRIMGS
                                44(R0)
                                LNK$GL_SHRSYMS
                                48(R0), 28(R1)
                                #64, 11(R1)
                                R1, 48(R0)
                                #1, R0
                                04 00042
                                51 00000000' EF D0 00002
                                50      20 A1 D0 00009
                                2E      58 A0 E8 0000D
                                27 00000000G 00 E8 00011
                                OA A1      C3 E1 00018
                                2C      A0 D5 0001D
                                06      12 00020
                                00000000G 00 D6 00022
                                2C      A0 D6 00028 1$:
                                00000000G 00 D6 0002B
                                1C A1      30 A0 D0 00031
                                OB A1      40 8F 88 00036
                                30 A0      51 D0 00038
                                50      01 D0 0003F 2$:
                                04 00042
                                RET
```

; Routine Size: 67 bytes, Routine Base: \$CODE\$ + 0C54

```
: 1546      2028  2  ! MAIN BODY OF PROSYMBOL
: 1547      2029  2  !
: 1548      2030  2  !
: 1549      2031  2  local
: 1550      2032  2      found,
: 1551      2033  2      symbolvalue,
: 1552      2034  2      symbolcluster : ref block [, byte],
: 1553      2035  2      ntxtsyment : ref block [, byte],
: 1554      2036  2      psctmapent : ref block [, byte],
: 1555      2037  2      psctdesc : ref block [, byte],
: 1556      2038  2      modpscontriblk : ref block [, byte];
: 1557      2039  2  !
: 1558      2040  2  bind
: 1559      2041  2      symbolrec = objvec [.gsdoffset] : block [, byte];
: 1560      2042  2  builtin
: 1561      2043  2      insque,
: 1562      2044  2      remque;
: 1563      2045  2
: 1564      2046  2  symtabent = 0;
: 1565      2047  2  !
: 1566      2048  2  ! FIRST CHECK VALID SYMBOL NAME
: 1567      2049  2  !
: 1568      2050  2  !
: 1569      2051  2  !
: 1570      2052  2  if .symbolstring [0] gtru sym$c_maxlng
: 1571      2053  2  or .symbolstring [0] eql 0
: 1572      2054  3  then begin
                                ! IF THE SYMBOL LENGTH IS OUTSIDE
                                ! LEGAL RANGE
```

```
: 1573      2055      3      signal (lin$_illnamelen, 0, lnk$gt_symstring
: 1574      2056      3      ,symbolstring [0], .symbolstring [0]
: 1575      2057      3      ,sym$$_maxlng, obmodesc [omd$b_namlng]
: 1576      2058      3      ,lnk$gt_curfil [fdb$q_filename]
: 1577      2059      3      );
: 1578      2060      3      return false;
: 1579      2061      3      end;
: 1580      2062      3
: 1581      2063      3      symbolvalue = (if .symbolrec [gsy$v_def]
: 1582      2064      4      then (if not .lclsymgsd
: 1583      2065      5      then (if .wordpsectgsd
: 1584      2066      5      then .symbolrec [sdfw$_value]
: 1585      2067      5      else .symbolrec [sdf$_value]
: 1586      2068      5      )
: 1587      2069      4      else .symbolrec [lsdf$_value]
: 1588      2070      4      )
: 1589      2071      3      else 0
: 1590      2072      3      );
: 1591      2073      2
: 1592      2074      4      if not (found = (if .lclsymgsd
: 1593      2075      4      then lnk$searchlocal (symbolstring [0], .symbolrec [lsy$_envindx], symtabent, symentnam)
: 1594      2076      4      else lnk$search (symbolstring [0], symtabent, symentnam)
: 1595      2077      3      )
: 1596      2078      2      then if .obmodesc [omd$v_selser] and .symbolrec [gsy$v_def]
: 1597      2079      2      then return true
: 1598      2080      2      else begin
: 1599      2081      3      lnk$insert (symbolstring [0], symtabent, symentnam);
: 1600      2082      3      if .lclsymgsd
: 1601      2083      3      then lnk$gw_nlsyms = .lnk$gw_nlsyms + 1;
: 1602      2084      3      else lnk$gw_nsymbols = .lnk$gw_nsymbols + 1;
: 1603      2085      3
: 1604      2086      3      symtabent [sym$v_weak] = .symbolrec [gsy$v_weak];
: 1605      2087      3      symtabent [sym$v_rel] = .symbolrec [gsy$v_rel];
: 1606      2088      3      symtabent [sym$v_lclsym] = .lclsymgsd;
: 1607      2089      3
: 1608      2090      3      if .lnk$gl_ctlmsk [lnk$v_intfil]
: 1609      2091      3      and
: 1610      2092      3      .lnk$gl_ctlmsk [lnk$v_supsys]
: 1611      2093      3      or .symtabent [sym$v_lclsym]
: 1612      2094      3      then symtabent [sym$v_supres] = true;
: 1613      2095      3
: 1614      2096      3      if not .symbolrec [gsy$v_def]
: 1615      2097      4      then begin
: 1616      2098      4      lnk$insudfsym (.symtabent);
: 1617      2099      4      if not .symtabent [sym$v_weak]
: 1618      2100      4      and not .symtabent [sym$v_lclsym]
: 1619      2101      5      then begin
: 1620      2102      5      if .lnk$gl_libsym neq 0
: 1621      2103      6      then begin
: 1622      2104      6
: 1623      2105      6      IF THE CURRENT FILE IS A SEARCH LIBRARY AND THE MODULE IS PULLED
: 1624      2106      6      AND THE LIST HAS EMPTIED DURING THE PROCESSING OF THE MODULE OR
: 1625      2107      6      FOR THE FILE SO THAT THE LIBRARY SEARCH WILL RETURN TO THE TOP OF
: 1626      2108      6      UNDEFINED LIST ONE MORE TIME.
: 1627      2109      6
: 1628      2110      6      bind libsymnam = .lnk$gl_libsym - .lnk$gl_libsym [sym$b_namlng]
: 1629      2111      6      - snb$_fxdlen : block [, byte];
```



```
: 1630      2112  6
: 1631      2113  6
: 1632      2114  6
: 1633      2115  7
: 1634      2116  7
: 1635      2117  7
: 1636      2118  7
: 1637      2119  7
: 1638      2120  6
: 1639      2121  5
: 1640      2122  4
: 1641      2123  4
: 1642      2124  4
: 1643      2125  4
: 1644      2126  3
: 1645      2127  2
: 1646      2128  2
: 1647      2129  2
: 1648      2130  2
: 1649      2131  3
: 1650      2132  3
: 1651      2133  3
: 1652      2134  4
: 1653      2135  4
: 1654      2136  4
: 1655      2137  5
: 1656      2138  5
: 1657      2139  5
: 1658      2140  5
: 1659      2141  5
: 1660      2142  5
: 1661      2143  5
: 1662      2144  5
: 1663      2145  5
: 1664      2146  5
: 1665      2147  5
: 1666      2148  5
: 1667      2149  5
: 1668      2150  5
: 1669      2151  5
: 1670      2152  5
: 1671      2153  5
: 1672      2154  5
: 1673      2155  5
: 1674      2156  5
: 1675      2157  5
: 1676      2158  5
: 1677      2159  5
: 1678      2160  5
: 1679      2161  4
: 1680      2162  4
: 1681      2163  4
: 1682      2164  4
: 1683      2165  4
: 1684      2166  4
: 1685      2167  4
: 1686      2168  4

      if .lnk$gl_curfil [fdb$v_libsrch]
      and
      (.lnk$gl_libsym eql lnk$gl_udflst
      or
      ch$ls (.symntnam [snb$b_namlng], symntnam [snb$t_name]
      )
      .libsymnam [snb$b_namlng], libsymnam [snb$t_name]
      )
      then lnk$gl_curfil [fdb$v_newudf] = true;
      end;

      end;

      crefilter (false, .symbolrec [gsy$v_weak]);
      return true;
      end;

      end;

      ! AND WE ARE DONE WITH THIS
      ! UNRESOLVED REFERENCE SO RETURN SUCCESS

      if .symtabent [sym$v_def]
      then begin
      if .obmodesc [omd$v_selser]
      then return true
      else if not .symbolrec [gsy$v_def]
      then begin
      if .symtabent [sym$v_shring]
      and not .symtabent [sym$v_crosref]
      then begin
      local      defomdptr : ref block [, byte];

      if not .symtabent [sym$v_gref]
      then shringsym ();

      lib$lookup_tree ( lnk$gl_omdtree
      , .symtabent [sym$l_omdnum]
      , lnk$compare_omd
      , defomdptr
      );

      crefilter ( true, .symtabent [sym$v_weak]
      , .defomdptr [node$l_ptr]
      );

      if .lnk$gl_ctlmsk [lnk$v_long]
      and not .symtabent [sym$v_rel]
      and not .symtabent [sym$v_supres]
      then crf$insrtref ( lnk$al_valctltb
      , symtabent [sym$l_value]
      , symntnam [snb$b_namlng]
      , .symtabent [sym$w_flags]
      , 0
      );

      end;

      crefilter (false, .symbolrec [gsy$v_weak]);

      if .symtabent [sym$v_shring]
      and not .symtabent [sym$v_gref]
      then shringsym ();

      ! IF THE ENTRY IN
      ! THE SYMBOL TABLE IS
      ! A DEFINITION AND
      ! THIS IS SELECTIVE
      ! SEARCH, IGNORE SYMBOL.
      ! IGNORE REFERENCE TO DEFINED SYMBOL EXCEPT FOR CREF
      ! IF SYMBOL IS FROM SHAREABLE IMAGE
      ! AND HAS NOT BEEN CROSS-REF'D YET
      ! ENTER SYMBOL INTO SHR LIST IF NOT DONE YET
      ! FIND OMD FOR MODULE THIS SYMBOL DEFINED IN
      ! CROSS REF THE DEFINITION
      ! IF A LONG FORM MAP IS REQUIRED
      ! AND THIS SYMBOL IS BASED
      ! AND NOT SUPRESSED
      ! INSERT THIS
      ! SYMBOL AS A REFERENCE TO ITS
      ! VALUE
      ! IF SYMBOL IS FROM SHAREABLE IMAGE
      ! AND HAS NOT BEEN LINKED INTO SHR LIST
      ! THEN DO SO NOW
```



```

: 1687      2169      4      return true;
: 1688      2170      4      end;
: 1689      2171      4
: 1690      2172      4
: 1691      2173      4      !
: 1692      2174      4      ! HERE WE HAVE A MULTIPLE DEFINITION FROM A NON SELECTIVE SEARCH MODULE.
: 1693      2175      4      ! IF BOTH DEFINITIONS ARE ABSOLUTE AND THE VALUES ARE EQUAL ALL IS WELL.
: 1694      2176      4      ! OTHERWISE WE HAVE AN ERROR.
: 1695      2177      4      !
: 1696      2178      4
: 1697      2179      4      if .symbolvalue eql .syntabent [sym$l_value]
: 1698      2180      4      and not .symbolrec [gsy$v_rel]
: 1699      2181      4      and not .syntabent [sym$v_rel]
: 1700      2182      4      then return true;
: 1701      2183      4
: 1702      2184      4      if .syntabent [sym$v_optsym]
: 1703      2185      4      and not .symbolrec [gsy$v_rel]
: 1704      2186      4      then begin
: 1705      2187      6          psctmapent = fndpsctmapent ((if not .lclsymgsd
: 1706      2188      6              then (if .wordpsectgsd
: 1707      2189      6                  then .symbolrec [sdfw$w_psindx]
: 1708      2190      6                  else .symbolrec [sdf$b_psindx]
: 1709      2191      5              )
: 1710      2192      4              else .symbolrec [lsdf$w_psindx]
: 1711      2193      4          ));
: 1712      2194      4          psctdesc = .psctmapent [pmt$l_pscdes];
: 1713      2195      4          ! GET ADDRESS OF PSECT DESCRIPTOR
: 1714      2196      5          if .psctdesc neq 0
: 1715      2197      5          then begin
: 1716      2198      5              symtabent [sym$l_pscfst] = .psctdesc [psc$l_symlst];
: 1717      2199      5              psctdesc [psc$l_symlst] = .syntabent;
: 1718      2200      5              ! IF P-SECTION IS
: 1719      2201      5              ! DEFINED, GET THIS
: 1720      2202      5              ! P-SECT BASE
: 1721      2203      4              ! LINK SYMBOL
: 1722      2204      4              ! LIST
: 1723      2205      4              ! P-SECTION NOT
: 1724      2206      4              ! DEFINED YET, SO
: 1725      2207      3              ! LINK SYMBOL
: 1726      2208      3              ! MAPPING TABLE
: 1727      2209      4              else begin
: 1728      2210      4                  symtabent [sym$l_pscfst] = .psctmapent [pmt$l_symlst];
: 1729      2211      4                  psctmapent [pmt$l_symlst] = .syntabent;
: 1730      2212      4                  ! NOW WE CAN LEAVE QUIETLY...
: 1731      2213      3              end;
: 1732      2214      3          return true;
: 1733      2215      4          end;
: 1734      2216      4
: 1735      2217      4      if .syntabent [sym$v_shring]
: 1736      2218      4      and (if (symbolcluster = .syntabent [sym$l_cludsc]) eql 0
: 1737      2219      4      then false
: 1738      2220      4      else .symbolcluster [clu$v_based]
: 1739      2221      4      )
: 1740      2222      4      and .lnk$gl_curclu [clu$v_shring]
: 1741      2223      4      and .lnk$gl_curclu [clu$v_based]
: 1742      2224      4      and (.syntabent [sym$l_value] + .symbolcluster [clu$l_base]
: 1743      2225      3          eql
: 1744      2226      3          .symbolvalue + .lnk$gl_curclu [clu$l_base]
: 1745      2227      3      )
: 1746      2228      3      then return true;
: 1747      2229      3      ! THEN IGNORE THIS DEFINITION COMPLETE
: 1748      2230      3
: 1749      2231      3      signal (lin$_muldef, 3, symbolstring [0], obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
: 1750      2232      3
: 1751      2233      3      if not .symbolrec [gsy$v_rel]
: 1752      2234      3      and not .syntabent [sym$v_rel]
: 1753      2235      3      then symtabent [sym$l_value] = .symbolvalue;
: 1754      2236      3      ! IF BOTH ARE ABSOLUTE
: 1755      2237      3      ! THEN WE SIMPLY RE-DEFINE
: 1756      2238      3      ! THE VALUE
: 1757      2239      3
: 1758      2240      3
: 1759      2241      3
: 1760      2242      3
: 1761      2243      3
: 1762      2244      3
: 1763      2245      3
: 1764      2246      3
: 1765      2247      3
: 1766      2248      3
: 1767      2249      3
: 1768      2250      3
: 1769      2251      3
: 1770      2252      3
: 1771      2253      3
: 1772      2254      3
: 1773      2255      3
: 1774      2256      3
: 1775      2257      3
: 1776      2258      3
: 1777      2259      3
: 1778      2260      3
: 1779      2261      3
: 1780      2262      3
: 1781      2263      3
: 1782      2264      3
: 1783      2265      3
: 1784      2266      3
: 1785      2267      3
: 1786      2268      3
: 1787      2269      3
: 1788      2270      3
: 1789      2271      3
: 1790      2272      3
: 1791      2273      3
: 1792      2274      3
: 1793      2275      3
: 1794      2276      3
: 1795      2277      3
: 1796      2278      3
: 1797      2279      3
: 1798      2280      3
: 1799      2281      3
: 1800      2282      3
: 1801      2283      3
: 1802      2284      3
: 1803      2285      3
: 1804      2286      3
: 1805      2287      3
: 1806      2288      3
: 1807      2289      3
: 1808      2290      3
: 1809      2291      3
: 1810      2292      3
: 1811      2293      3
: 1812      2294      3
: 1813      2295      3
: 1814      2296      3
: 1815      2297      3
: 1816      2298      3
: 1817      2299      3
: 1818      2300      3
: 1819      2301      3
: 1820      2302      3
: 1821      2303      3
: 1822      2304      3
: 1823      2305      3
: 1824      2306      3
: 1825      2307      3
: 1826      2308      3
: 1827      2309      3
: 1828      2310      3
: 1829      2311      3
: 1830      2312      3
: 1831      2313      3
: 1832      2314      3
: 1833      2315      3
: 1834      2316      3
: 1835      2317      3
: 1836      2318      3
: 1837      2319      3
: 1838      2320      3
: 1839      2321      3
: 1840      2322      3
: 1841      2323      3
: 1842      2324      3
: 1843      2325      3
: 1844      2326      3
: 1845      2327      3
: 1846      2328      3
: 1847      2329      3
: 1848      2330      3
: 1849      2331      3
: 1850      2332      3
: 1851      2333      3
: 1852      2334      3
: 1853      2335      3
: 1854      2336      3
: 1855      2337      3
: 1856      2338      3
: 1857      2339      3
: 1858      2340      3
: 1859      2341      3
: 1860      2342      3
: 1861      2343      3
: 1862      2344      3
: 1863      2345      3
: 1864      2346      3
: 1865      2347      3
: 1866      2348      3
: 1867      2349      3
: 1868      2350      3
: 1869      2351      3
: 1870      2352      3
: 1871      2353      3
: 1872      2354      3
: 1873      2355      3
: 1874      2356      3
: 1875      2357      3
: 1876      2358      3
: 1877      2359      3
: 1878      2360      3
: 1879      2361      3
: 1880      2362      3
: 1881      2363      3
: 1882      2364      3
: 1883      2365      3
: 1884      2366      3
: 1885      2367      3
: 1886      2368      3
: 1887      2369      3
: 1888      2370      3
: 1889      2371      3
: 1890      2372      3
: 1891      2373      3
: 1892      2374      3
: 1893      2375      3
: 1894      2376      3
: 1895      2377      3
: 1896      2378      3
: 1897      2379      3
: 1898      2380      3
: 1899      2381      3
: 1900      2382      3
: 1901      2383      3
: 1902      2384      3
: 1903      2385      3
: 1904      2386      3
: 1905      2387      3
: 1906      2388      3
: 1907      2389      3
: 1908      2390      3
: 1909      2391      3
: 1910      2392      3
: 1911      2393      3
: 1912      2394      3
: 1913      2395      3
: 1914      2396      3
: 1915      2397      3
: 1916      2398      3
: 1917      2399      3
: 1918      2400      3
: 1919      2401      3
: 1920      2402      3
: 1921      2403      3
: 1922      2404      3
: 1923      2405      3
: 1924      2406      3
: 1925      2407      3
: 1926      2408      3
: 1927      2409      3
: 1928      2410      3
: 1929      2411      3
: 1930      2412      3
: 1931      2413      3
: 1932      2414      3
: 1933      2415      3
: 1934      2416      3
: 1935      2417      3
: 1936      2418      3
: 1937      2419      3
: 1938      2420      3
: 1939      2421      3
: 1940      2422      3
: 1941      2423      3
: 1942      2424      3
: 1943      2425      3
: 1944      2426      3
: 1945      2427      3
: 1946      2428      3
: 1947      2429      3
: 1948      2430      3
: 1949      2431      3
: 1950      2432      3
: 1951      2433      3
: 1952      2434      3
: 1953      2435      3
: 1954      2436      3
: 1955      2437      3
: 1956      2438      3
: 1957      2439      3
: 1958      2440      3
: 1959      2441      3
: 1960      2442      3
: 1961      2443      3
: 1962      2444      3
: 1963      2445      3
: 1964      2446      3
: 1965      2447      3
: 1966      2448      3
: 1967      2449      3
: 1968      2450      3
: 1969      2451      3
: 1970      2452      3
: 1971      2453      3
: 1972      2454      3
: 1973      2455      3
: 1974      2456      3
: 1975      2457      3
: 1976      2458      3
: 1977      2459      3
: 1978      2460      3
: 1979      2461      3
: 1980      2462      3
: 1981      2463      3
: 1982      2464      3
: 1983      2465      3
: 1984      2466      3
: 1985      2467      3
: 1986      2468      3
: 1987      2469      3
: 1988      2470      3
: 1989      2471      3
: 1990      2472      3
: 1991      2473      3
: 1992      2474      3
: 1993      2475      3
: 1994      2476      3
: 1995      2477      3
: 1996      2478      3
: 1997      2479      3
: 1998      2480      3
: 1999      2481      3
: 2000      2482      3
: 2001      2483      3
: 2002      2484      3
: 2003      2485      3
: 2004      2486      3
: 2005      2487      3
: 2006      2488      3
: 2007      2489      3
: 2008      2490      3
: 2009      2491      3
: 2010      2492      3
: 2011      2493      3
: 2012      2494      3
: 2013      2495      3
: 2014      2496      3
: 2015      2497      3
: 2016      2498      3
: 2017      2499      3
: 2018      2500      3
: 2019      2501      3
: 2020      2502      3
: 2021      2503      3
: 2022      2504      3
: 2023      2505      3
: 2024      2506      3
: 2025      2507      3
: 2026      2508      3
: 2027      2509      3
: 2028      2510      3
: 2029      2511      3
: 2030      2512      3
: 2031      2513      3
: 2032      2514      3
: 2033      2515      3
: 2034      2516      3
: 2035      2517      3
: 2036      2518      3
: 2037      2519      3
: 2038      2520      3
: 2039      2521      3
: 2040      2522      3
: 2041      2523      3
: 2042      2524      3
: 2043      2525      3
: 2044      2526      3
: 2045      2527      3
: 2046      2528      3
: 2047      2529      3
: 2048      2530      3
: 2049      2531      3
: 2050      2532      3
: 2051      2533      3
: 2052      2534      3
: 2053      2535      3
: 2054      2536      3
: 2055      2537      3
: 2056      2538      3
: 2057      2539      3
: 2058      2540      3
: 2059      2541      3
: 2060      2542      3
: 2061      2543      3
: 2062      2544      3
: 2063      2545      3
: 2064      2546      3
: 2065      2547      3
: 2066      2548      3
: 2067      2549      3
: 2068      2550      3
: 2069      2551      3
: 2070      2552      3
: 2071      2553      3
: 2072      2554      3
: 2073      2555      3
: 2074      2556      3
: 2075      2557      3
: 2076      2558      3
: 2077      2559      3
: 2078      2560      3
: 2079      2561      3
: 2080      2562      3
: 2081      2563      3
: 2082      2564      3
: 2083      2565      3
: 2084      2566      3
: 2085      2567      3
: 2086      2568      3
: 2087      2569      3
: 2088      2570      3
: 2089      2571      3
: 2090      2572      3
: 2091      2573      3
: 2092      2574      3
: 2093      2575      3
: 2094      2576      3
: 2095      2577      3
: 2096      2578      3
: 2097      2579      3
: 2098      2580      3
: 2099      2581      3
: 2100      2582      3
: 2101      2583      3
: 2102      2584      3
: 2103      2585      3
: 2104      2586      3
: 2105      2587      3
: 2106      2588      3
: 2107      2589      3
: 2108      2590      3
: 2109      2591      3
: 2110      2592      3
: 2111      2593      3
: 2112      2594      3
: 2113      2595      3
: 2114      2596      3
: 2115      2597      3
: 2116      2598      3
: 2117      2599      3
: 2118      2600      3
: 2119      2601      3
: 2120      2602      3
: 2121      2603      3
: 2122      2604      3
: 2123      2605      3
: 2124      2606      3
: 2125      2607      3
: 2126      2608      3
: 2127      2609      3
: 2128      2610      3
: 2129      2611      3
: 2130      2612      3
: 2131      2613      3
: 2132      2614      3
: 2133      2615      3
: 2134      2616      3
: 2135      2617      3
: 2136      2618      3
: 2137      2619      3
: 2138      2620      3
: 2139      2621      3
: 2140      2622      3
: 2141      2623      3
: 2142      2624      3
: 2143      2625      3
: 2144      2626      3
: 2145      2627      3
: 2146      2628      3
: 2147      2629      3
: 2148      2630      3
: 2149      2631      3
: 2150      2632      3
: 2151      2633      3
: 2152      2634      3
: 2153      2635      3
: 2154      2636      3
: 2155      2637      3
: 2156      2638      3
: 2157      2639      3
: 2158      2640      3
: 2159      2641      3
: 2160      2642      3
: 2161      2643      3
: 2162      2644      3
: 2163      2645      3
: 2164      2646      3
: 2165      2647      3
: 2166      2648      3
: 2167      2649      3
: 2168      2650      3
: 2169      2651      3
: 2170      2652      3
: 2171      2653      3
: 2172      2654      3
: 2173      2655      3
: 2174      2656      3
: 2175      2657      3
: 2176      2658      3
: 2177      2659      3
: 2178      2660      3
: 2179      2661      3
: 2180      2662      3
: 2181      2663      3
: 2182      2664      3
: 2183      2665      3
: 2184      2666      3
: 2185      2667      3
: 2186      2668      3
: 2187      2669      3
: 2188      2670      3
: 2189      2671      3
: 2190      2672      3
: 2191      2673      3
: 2192      2674      3
: 2193      2675      3
: 2194      2676      3
: 2195      2677      3
: 2196      2678      3
: 2197      2679      3
: 2198      2680      3
: 2199      2681      3
: 2200      2682      3
: 2201      2683      3
: 2202      2684      3
: 2203      2685      3
: 2204      2686      3
: 2205      2687      3
: 2206      2688      3
: 2207      2689      3
: 2208      2690      3
: 2209      2691      3
: 2210      2692      3
: 2211      2693      3
: 2212      2694      3
: 2213      2695      3
: 2214      2696      3
: 2215      2697      3
: 2216      2698      3
: 2217      2699      3
: 2218      2700      3
: 2219      2701      3
: 2220      2702      3
: 2221      2703      3
: 2222      2704      3
: 2223      2705      3
: 2224      2706      3
: 2225      2707      3
: 2226      2708      3
: 2227      2709      3
: 2228      2710      3
: 2229      2711      3
: 2230      2712      3
: 2231      2713      3
: 2232      2714      3
: 2233      2715      3
: 2234      2716      3
: 2235      2717      3
: 2236      2718      3
: 2237      2719      3
: 2238      2720      3
: 2239      2721      3
: 2240      2722      3
: 2241      2723      3
: 2242      2724      3
: 2243      2725      3
: 2244      2726      3
: 2245      2727      3
: 2246      2728      3
: 2247      2729      3
: 2248      2730      3
: 2249      2731      3
: 2250      2732      3
: 2251      2733      3
: 2252      2734      3
: 2253      2735      3
: 2254      2736      3
: 2255      2737      3
: 2256      2738      3
: 2257      2739      3
: 2258      2740      3
: 2259      2741      3
: 2260      2742      3
: 2261      2743      3
: 2262      2744      3
: 2263      2745      3
: 2264      2746      3
: 2265      2747      3
: 2266      2748      3
: 2267      2749      3
: 2268      2750      3
: 2269      2751      3
: 2270      2752      3
: 2271      2753      3
: 2272      2754      3
: 2273      2755      3
: 2274      2756      3
: 2275      2757      3
: 2276      2758      3
: 2277      2759      3
: 2278      2760      3
: 2279      2761      3
: 2280      2762      3
: 2281      2763      3
: 2282      2764      3
: 2283      2765      3
: 2284      2766      3
: 2285      2767      3
: 2286      2768      3
: 2287      2769      3
: 2288      2770      3
: 2289      2771      3
: 2290      2772      3
: 2291      2773      3
: 2292      2774      3
: 2293      2775      3
: 2294      2776      3
: 2295      2777      3
: 2296      2778      3
: 2297      2779      3
: 2298      2780      3
: 2299      2781      3
: 2300      2782      3
: 2301      2783      3
: 2302      2784      3
: 2303      2785      3
: 2304      2786      3
: 2305      2787      3
: 2306      2788      3
: 2307      2789      3
: 2308      2790      3
: 2309      2791      3
: 2310      2792      3
: 2311      2793      3
: 2312      2794      3
: 2313      2795      3
: 2314      2796      3
: 2315      2797      3
: 2316      2798      3
: 2317      2799      3
: 2318      2800      3
: 2319      2801      3
: 2320      2802      3
: 2321      2803      3
: 2322      2804      3
: 2323      2805      3
: 2324      2806      3
: 2325      2807      3
: 2326      2808      3
: 2327      2809      3
: 2328      2810      3
: 2329      2811      3
: 2330      2812      3
: 2331      2813      3
: 2332      2814      3
: 2333      2815      3
: 2334      2816      3
: 2335      2817      3
: 2336      2818      3
: 2337      2819      3
: 2338      2820      3
: 2339      2821      3
: 2340      2822      3
: 2341      2823      3
: 2342      2824      3
: 2343      2825      3
: 2344      2826      3
: 2345      2827      3
: 2346      2828      3
: 2347      2829      3
: 2348      2830      3
: 2349      2831      3
: 2350      2832      3
: 2351      2833      3
: 2352      2834      3
: 2353      2835      3
: 2354      2836      3
: 2355      2837      3
: 2356      2838      3
: 2357      2839      3
: 2358      2840      3
: 2359      2841      3
: 2360      2842      3
: 2361      2843      3
: 2362      2844      3
: 2363      2845      3
: 2364      2846      3
: 2365      2847      3
: 2366      2848      3
: 2367      2849      3
: 2368      2850      3
: 2369      2851      3
: 2370      2852      3
: 2371      2853      3
: 2372      2854      3
: 2373      2855      3
: 2374      2856      3
: 2375      2857      3
: 2376      2858      3
: 2377      2859      3
: 2378      2860      3
: 2379      2861      3
: 2380      2862      3
: 2381      2863      3
: 2382      2864      3
: 2383      2865      3
: 2384      2866      3
: 2385      2867      3
: 2386      2868      3
: 2387      2869      3
: 2388      2870      3
: 2389      2871      3
: 2390      2872      3
: 2391      2873      3
: 2392      2874      3
: 2393      2875      3
: 2394      2876      3
: 2395      2877      3
: 2396      2878      3
: 2397      2879      3
: 2398      2880      3
: 2399      2881      3
: 2400      2882      3
: 2401      2883      3
: 2402      2884      3
: 2403      2885      3
: 2404      2886      3
: 2405      2887      3
```



```

: 1744      2226      3
: 1745      2227      3
: 1746      2228      3
: 1747      2229      3
: 1748      2230      3
: 1749      2231      3
: 1750      2232      3
: 1751      2233      3
: 1752      2234      3
: 1753      2235      3
: 1754      2236      3
: 1755      2237      3
: 1756      2238      3
: 1757      2239      3
: 1758      2240      3
: 1759      2241      3
: 1760      2242      3
: 1761      2243      3
: 1762      2244      3
: 1763      2245      3
: 1764      2246      3
: 1765      2247      3
: 1766      2248      3
: 1767      2249      3
: 1768      2250      3
: 1769      2251      3
: 1770      2252      4
: 1771      2253      4
: 1772      2254      4
: 1773      2255      4
: 1774      2256      4
: 1775      2257      4
: 1776      2258      4
: 1777      2259      5
: 1778      2260      5
: 1779      2261      5
: 1780      2262      5
: 1781      2263      5
: 1782      2264      5
: 1783      2265      4
: 1784      2266      4
: 1785      2267      3
: 1786      2268      3
: 1787      2269      3
: 1788      2270      3
: 1789      2271      3
: 1790      2272      3
: 1791      2273      4
: 1792      2274      4
: 1793      2275      4
: 1794      2276      4
: 1795      2277      4
: 1796      2278      4
: 1797      2279      4
: 1798      2280      4
: 1799      2281      4
: 1800      2282      4

      return true;
      end
    else
      THE ENTRY IN THE SYMBOL TABLE IS EITHER A NEW REFERENCE
      JUST INSERTED OR A REFERENCE AWAITING DEFINITION.

      IF THIS IS AN INCOMING DEFINITION WE PROCEED TO DEFINE THE
      SYMBOL. THAT IS:
          IF UDFLINK FIELD IS NON ZERO, THE TABLE ENTRY
          IS ON THE UNDEFINED SYMBOL LIST, SO REMOVE IT
          AND DECREMENT THE UNDEFINED COUNT (PROVIDED IT IS
          NOT A WEAK REFERENCE).

          THEN COPY INTO THE SYMBOL TABLE ENTRY THE FLAGS,
          VALUE, ENTRY MASK, DATA TYPE AND FINALLY
          LINK THE SYMBOL ON THE OWNING P-SECTION LIST

      IF THIS IS (ANOTHER) INCOMING REFERENCE, AND IT IS
      NOT A WEAK REFERENCE, ENSURE THAT THE SYMBOL TABLE ENTRY IS
      ALSO NOT A WEAK REFERENCE AND THAT THIS STRONG REFERENCE
      TO AN UNDEFINED SYMBOL IS COUNTED.

      begin
      if not .symbolrec [gsy$u_def]
      then begin
          crefilter (false, .symbolrec [gsy$u_weak]);

          if .symbolrec [gsy$u_weak]
          then return true;

          if .syntabent [sym$u_weak]
          then begin
              syntabent [sym$u_weak] = false;

              if .syntabent [sym$u_lclsym]
              then lnk$gw_nudflsyms = .lnk$gw_nudflsyms + 1
              else lnk$gw_nudfsyms = .lnk$gw_nudfsyms + 1;
              end;

          return true;
          end;

          ! IF THIS IS
          ! ANOTHER REFERENCE THEN
          ! CROSS REFERENCE IT

          ! IF WEAK, IGNORE IT SINCE THE ONE WE
          ! HAVE IS AT LEAST WEAK AND NOTHING TO
          ! BE DONE
          ! IF A STRONG REFERENCE AND SYMBOL TABLE
          ! ENTRY IS WEAK
          ! MAKE SURE IT IS STRONG

          ! AND COUNT THE STRONG REFERENCE

          ! AND THAT'S IT FOR REFERENCES.

          ! IF THIS SYMBOL TABLE ENTRY IS ON
          ! UNDEFINED SYMBOL LIST,
          ! REMOVE FROM THE LIST

          ! IF THE ONE REMOVED IS THE ONE TO SEARCH
          ! FOR NEXT, MOVE TO ONE BEYOND

          ! IF IT IS NOT A WEAK REFERENCE

      AND NOW FOR A DEFINITION
      if .syntabent [sym$l_udflink] neq 0
      then begin
          remque (syntabent [sym$l_udflink], nextsyment);

          if .nextsyment eql .lnk$gl_libsym
          then lnk$gl_libsym = .nextsyment [sym$l_udflink];

          if not .syntabent [sym$u_weak]
          then
              if .syntabent [sym$u_lclsym]
              then lnk$gw_nudflsyms = .lnk$gw_nudflsyms - 1

```



```
1801 2283 4      else lnk$gw_nudfsyms = .lnk$gw_nudfsyms - 1;          ! DECREMENT NUMBER UNDEFINED
1802 2284      end;
1803 2285
1804 2286      !
1805 2287      ! Allow the symbol table entry to inherit the suppress and local symbol flags, but ignore
1806 2288      ! the universal flag. If the symbol is to be universal, then it has either been
1807 2289      ! already declared that way (via the UNIVERSAL=name option), or will be set by a
1808 2290      ! redefinition later (i.e., a transfer vector). It is not clear what an incoming
1809 2291      ! universal symbol (ref OR def) means.
1810 2292      !
1811 2293
1812 2294      symtabent [sym$w_flags] = (.symtabent [sym$w_flags] and (sym$m_lclsym or sym$m_supres or sym$m_uni))
1813 2295      or
1814 2296      (.symbolrec [gsy$w_flags] and (not sym$m_uni));
1815 2297
1816 2298      symtabent [sym$v_entmsk] = .entmskflg;          ! COPY THE ENTRY MASK FLAG
1817 2299
1818 2300      if .lnk$gl_ctlmsk [lnk$v_alluniv]          ! If all globals are to be promoted
1819 2301      then symtabent[sym$v_uni] = true;          ! to universal, do this one now
1820 2302
1821 2303      symtabent [sym$b_datyp] = .symbolrec [gsy$b_datyp];          ! COPY DATA TYPE
1822 2304      symtabent [sym$l_value] = .symbolvalue;          ! COPY THE VALUE
1823 2305      symtabent [sym$w_entmsk] = .entrymask;          ! COPY THE MASK
1824 2306      symtabent [sym$v_shring] = .obmodesc [omd$v_shring];          ! SET/CLEAR SHAREABLE IMAGE FLAG
1825 2307      psctmapent = fndpsctmapent ((if not .lclsymgsd
1826 2308      then (if .wordpsectgsd          ! FIND PSECT MAPPING TABLE ENTRY
1827 2309      then .symbolrec [sdfw$w_psindx]
1828 2310      else .symbolrec [sdf$b_psindx]
1829 2311      )
1830 2312      else .symbolrec [lsdf$w_psindx]
1831 2313      ));
1832 2314      psctdesc = .psctmapent [pmt$l_pscdes];          ! GET ADDRESS OF PSECT DESCRIPTOR
1833 2315      symtabent [sym$l_cludsc] = .lnk$gl_curclu;          ! SET POINTER TO CLUSTER DESCRIPTOR
1834 2316      symtabent [sym$l_omdnum] = .lnk$gw_nmodules;          ! SET INDEX OF DEFINING MODULE
1835 2317
1836 2318      if .symtabent [sym$v_shring]          ! COPY SHR IMAGE FLAG
1837 2319      then begin
1838 2320      symtabent [sym$v_uni] = false;          ! CLEARING UNI IF SYMBOL FROM ANOTHER IMAGE
1839 2321
1840 2322      if .found and not .symtabent [sym$v_gref]          ! AND HAS NOT BEEN LINKED INTO THE LIST YET
1841 2323      then shringsym ();          ! THEN DO IT NOW
1842 2324
1843 2325      ! DON'T CREF SHAREABLE IMAGE SYMBOLS ON DEF. IT WILL
1844 2326      ! BE CREF'ED WHEN A REFERENCE TO IT IS ENCOUNTERED
1845 2327      end
1846 2328      else
1847 2329      lnk$gw_lsymbols = .lnk$gw_lsymbols + 1;          ! NOT FROM A SHR IMAGE,
1848 2330      ! COUNT ANOTHER SYMBOL THIS IMAGE
1849 2331
1850 2332      if .found          ! IF SYMBOL ALREADY REFERENCED
1851 2333      or not .symtabent [sym$v_shring]          ! OR NOT FROM A SHAREABLE IMAGE
1852 2334      then begin
1853 2335      crefilter (true, .symbolrec [gsy$v_weak]);          ! SO CROSS-REFERENCE IT NOW
1854 2336
1855 2337      if .lnk$gl_ctlmsk [lnk$v_long]          ! IF A LONG FORM MAP IS REQUIRED
1856 2338      and not .symtabent [sym$v_rel]          ! AND SYMBOL IS ABSOLUTE
1857 2339      and not .symtabent [sym$v_supres]          ! AND NOT SUPRESSED
1858 2340      then
```



```
: 1858      2340  4      crf$insrtref ( lnk$al_valctltb      | INSERT THIS
: 1859      2341  4      , symtabent [sym$l_value]      | SYMBOL AS A REFERENCE TO ITS
: 1860      2342  4      , symentnam [snb$b_namlng]      | VALUE
: 1861      2343  4      , symtabent [sym$w_flags]
: 1862      2344  4      ;
: 1863      2345  4      end;
: 1864      2346  3
: 1865      2347  3
: 1866      2348  3      if .psctdesc neq 0
: 1867      2349  4      then begin
: 1868      2350  4          modpscontriblk = .psctmapent [pmt$l_modcon];
: 1869      2351  4          symtabent [sym$l_value] = .symtabent [sym$l_value] +
: 1870      2352  4          .modpscontriblk [mpc$l_offset];
: 1871      2353  4          symtabent [sym$l_psc1st] = .psctdesc [psc$l_symlst];
: 1872      2354  4          psctdesc [psc$l_symlst] = .symtabent;
: 1873      2355  4          end
: 1874      2356  4      else begin
: 1875      2357  4          symtabent [sym$l_psc1st] = .psctmapent [pmt$l_symlst];
: 1876      2358  4          psctmapent [pmt$l_symlst] = .symtabent;
: 1877      2359  3          end;
: 1878      2360  3
: 1879      2361  3      return true;
: 1880      2362  2      end;
: 1881      2363  2
: 1882      2364  1 end;

! END OF DEFINITION
! END OF SYMBOL ROUTINE
```

```
OFFC 00000 PROSYMBOL:
5B 00000000' EF 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 ; 1967
5A FECA CF 9E 00009 MOVAB LNK$GW_NUDFLSYMS, R11
59 00000000' EF 9E 0000E MOVAB CREFILTER, R10
5E 04 C2 00015 MOVAB SYMTABENT, R9
55 EC A9 3C 00018 SUBL2 #4, SP
55 F8 A9 C0 0001C MOVZWL GSDOFFSET, R5
50 FC B9 9A 00022 ADDL2 OBJVEC, R5
1F 50 91 00026 CLRL SYMTABENT
04 1A 00029 MOVZBL @SYMBOLSTRING, R0
50 D5 0002B CMPB R0, #31
2A 12 0002D BGTRU 1$
7E 00000000G 00 14 C1 0002F 1$: TSTL R0
50 A9 9F 00037 BNEQ 2$
1F DD 0003A ADDL3 #20, LNK$GL_CURFIL, -(SP)
50 DD 0003C PUSHAB OBMODESC+40-
FC A9 DD 0003E PUSHL #31
00000000' EF 9F 00041 PUSHL R0
06 DD 00047 PUSHL SYMBOLSTRING
00000000G 8F DD 00049 PUSHAB LNK$GT_SYMSTRING
08 FB 0004F PUSHL #6
03D3 31 00056 PUSHL #LINS_ILLNAMELEN
57 02 A5 9E 00059 2$: CALLS #8, LIB$SIGNAL
67 01 E1 0005D BRW 54$
1A 10 E5 A9 E8 00061 MOVAB 2(R5), R7
BBC #1, (R7), 5$
BLBS LCLSYMGSD, 4$ ; 2064
```


	06	E6	A9	E9	00065	BLBC	WORDPSECTGSD, 3\$	:	2065	
	54	06	A5	D0	00069	MOVL	6(R5), SYMBOLVALUE	:	2066	
			0E	11	0006D	BRB	6\$	:		
	54	05	A5	D0	0006F	3\$: MOVL	5(R5), SYMBOLVALUE	:	2067	
			08	11	00073	BRB	6\$	:	2065	
	54	08	A5	D0	00075	4\$: MOVL	8(R5), SYMBOLVALUE	:	2069	
			02	11	00079	BRB	6\$	:	2064	
			54	D4	0007B	5\$: CLRL	SYMBOLVALUE	:	2063	
	15	E5	A9	E9	0007D	6\$: BLBC	LCLSYMGSD, 7\$	:	2074	
		04	A9	9F	00081	PUSHAB	SYMENTNAM	:	2075	
			59	DD	00084	PUSHL	R9	:		
	7E	04	A5	3C	00086	MOVZWL	4(R5), -(SP)	:		
		FC	A9	DD	0008A	PUSHL	SYMBOLSTRING	:		
	00000000G	00	04	FB	0008D	CALLS	#4, LNK\$SEARCHLOCAL	:		
			0F	11	00094	BRB	8\$	:		
		04	A9	9F	00096	7\$: PUSHAB	SYMENTNAM	:	2076	
			59	DD	00099	PUSHL	R9	:		
	00000000G	00	FC	A9	DD	0009B	PUSHL	SYMBOLSTRING	:	
		58	03	FB	0009E	CALLS	#3, LNK\$SEARCH	:		
	03		50	D0	000A5	8\$: MOVL	R0, FOUND	:	2074	
			58	E9	000A8	BLBC	FOUND, 9\$	:		
			00C2	31	000AB	BRW	19\$	:		
07	3C	A9	03	E1	000AE	9\$: BBC	#3, OBMODESC+20, 10\$	:	2078	
03		67	01	E1	000B3	BBC	#1, (R7), 10\$	:		
			036E	31	000B7	BRW	53\$	:		
			04	A9	9F	000BA	10\$: PUSHAB	SYMENTNAM	2081	
				59	DD	000BD	PUSHL	R9		
	00000000G	00	FC	A9	DD	000BF	PUSHL	SYMBOLSTRING		
		05	03	FB	000C2	CALLS	#3, LNK\$INSERT	:		
			E5	A9	E9	000C9	BLBC	LCLSYMGSD, 11\$	2082	
			FE	AB	B6	000CD	INCW	LNK\$GW_NLSYMS	2083	
				03	11	000D0	BRB	12\$		
			F6	AB	B6	000D2	11\$: INCW	LNK\$GW_NSymbols	2084	
		51		69	D0	000D5	12\$: MCVL	SYMTABENT, R1	2086	
		50	0A	A1	9E	000D8	MOVAB	10(R1), R0		
		00		67	F0	000DC	INSV	(R7), #0, #1, (R0)		
		01		03	EF	000E1	EXTZV	#3, #1, (R7), R2	2087	
		01		52	F0	000E6	INSV	R2, #3, #1, (R0)		
		01	E5	A9	F0	000EB	INSV	LCLSYMGSD, #0, #1, 1(R0)	2088	
	08	00000000G	00	03	E1	000F2	BBC	#3, LNK\$GL_CTLMSK+1, 13\$	2090	
	04	00000000G	00	06	E0	000FA	BBS	#6, LNK\$GL_CTLMSK+1, 14\$	2092	
			04	A0	E9	00102	13\$: BLBC	1(R0), 15\$	2093	
		01		20	88	00106	14\$: BISB2	#32, 1(R0)	2094	
	62			01	E0	0010A	15\$: BBS	#1, (R7), 19\$	2096	
				51	DD	0010E	PUSHL	R1	2098	
				01	FB	00110	CALLS	#1, LNK\$INSUDFSYM		
	F701	CA		69	D0	00115	MOVL	SYMTABENT, R0	2099	
		50		A0	E8	00118	BLBS	10(R0), 17\$		
		47	0A	A0	E8	0011C	BLBS	11(R0), 17\$	2100	
		43	0B	00	D0	00120	MOVL	LNK\$GL_LIBSYM, R0	2102	
		50	00000000G	3A	13	00127	BEQL	17\$		
		51		0F	A0	9A	00129	MOVZBL	15(R0), R1	2110
		50		51	C3	0012D	SUBL3	R1, R0, R1		
		51		05	C2	00131	SUBL2	#5, R1	2111	
		56	00000000G	00	D0	00134	MOVL	LNK\$GL_CURFIL, R6	2113	
				0A	A6	95	0013B	TSTB	10(R6)	
				23	18	0013E	BGEQ	17\$		

52	00	05	52	B6	AB	9E	00140	MOVAB	LNK\$GL_UDFLST, R2	2115
			52		50	D1	00144	CMPL	R0, R2	
			50	04	16	13	00147	BEQL	16\$	
			53	04	A9	D0	00149	MOVL	SYMENTNAM, R0	2117
			52	04	A0	9A	0014D	MOVZBL	4(R0), R3	
			52	04	A1	9A	00151	MOVZBL	4(R1), R2	2118
			A0	05	53	2D	00155	CMPC5	R3, 5(R0), #0, R2, 5(R1)	
					A1		0015B			
					04	1E	0015D	BGEQU	17\$	
7E	67	0A	A6		01	88	0015F	BISB2	#1, 10(R6)	2120
			01		00	EF	00163	EXTZV	#0, #1, (R7), -(SP)	2124
					7E	D4	00168	CLRL	-(SP)	
			6A		02	FB	0016A	CALLS	#2, CREFILTER	
					02B8	31	0016D	BRW	53\$	2125
			50		69	D0	00170	MOVL	SYMTABENT, R0	2129
	03	0A	A0		01	E0	00173	BBS	#1, 10(R0), 20\$	
					0157	31	00178	BRW	34\$	
	ED	3C	A9		03	E0	0017B	BBS	#3, OBMODESC+20, 18\$	2131
	03		67		01	E1	00180	BBC	#1, (R7), 21\$	2133
					0091	31	00184	BRW	25\$	
	6D	0B	A0		03	E1	00187	BBC	#3, 11(R0), 23\$	2135
	68	0C	A0		01	E0	0018C	BBS	#1, 12(R0), 23\$	2136
	05	0B	A0		06	E0	00191	BBS	#6, 11(R0), 22\$	2140
		00E6	CA		00	FB	00196	CALLS	#0, SHRIMGSYM	2141
					5E	DD	0019B	PUSHL	SP	2143
				FBOF	CA	9F	0019D	PUSHAB	LNK\$COMPARE OMD	
			50		69	DD	001A1	MOVL	SYMTABENT, R0	2144
				14	A0	DD	001A4	PUSHL	20(R0)	
				EA	AB	9F	001A7	PUSHAB	LNK\$GL_OMDTREE	2143
		00000000G	00		04	FB	001AA	CALLS	#4, LIB\$LOOKUP_TREE	
			50		6E	D0	001B1	MOVL	DEFOMDPTR, R0	2149
				0A	A0	DD	001B4	PUSHL	10(R0)	
7E	0A	A0	50		69	D0	001B7	MOVL	SYMTABENT, R0	2148
			01		00	EF	001BA	EXTZV	#0, #1, 10(R0), -(SP)	
					01	DD	001C0	PUSHL	#1	
			6A		03	FB	001C2	CALLS	#3, CREFILTER	
			2D	00000000G	00	E9	001C5	BLBC	LNK\$GL_CTLMSK+1, 23\$	2152
			50		69	D0	001CC	MOVL	SYMTABENT, R0	2153
	25	0A	A0		03	E0	001CF	BBS	#3, 10(R0), 23\$	
			50		69	D0	001D4	MOVL	SYMTABENT, R0	2154
	1D	0B	A0		05	E0	001D7	BBS	#5, 11(R0), 23\$	
					7E	D4	001DC	CLRL	-(SP)	2157
			50		69	D0	001DE	MOVL	SYMTABENT, R0	2158
			7E		A0	3C	001E1	MOVZWL	10(R0), -(SP)	
	7E	04	A9	0A	04	C1	001E5	ADDL3	#4, SYMENTNAM, -(SP)	2157
					50	DD	001EA	PUSHL	R0	
				00000000G	00	9F	001EC	PUSHAB	LNK\$AL_VALCTLTB	2155
			00		05	FB	001F2	CALLS	#5, CRF\$INSRTREF	2157
7E			01		00	EF	001F9	EXTZV	#0, #1, (R7), -(SP)	2163
					7E	D4	001FE	CLRL	-(SP)	
			6A		02	FB	00200	CALLS	#2, CREFILTER	
			50		69	D0	00203	MOVL	SYMTABENT, R0	2165
	0A	0B	A0		03	E1	00206	BBC	#3, 11(R0), 24\$	
	05	0B	A0		06	E0	0020B	BBS	#6, 11(R0), 24\$	2166
		00E6	CA		00	FB	00210	CALLS	#0, SHRIMGSYM	2167
					0210	31	00215	BRW	53\$	2169
			50		69	D0	00218	MOVL	SYMTABENT, R0	2178

		60		54	D1	0021B	CMPL	SYMBOLVALUE, (R0)		
				09	12	0021E	BNEQ	26\$		
05		67		03	E0	00220	BBS	#3, (R7), 26\$	2179	
EC	0A	A0		03	E1	00224	BBC	#3, 10(R0), 24\$	2180	
36	0B	A0		01	E1	00229	BBC	#1, 11(R0), 32\$	2183	
32		67		03	E0	0022E	BBS	#3, (R7), 32\$	2184	
		12	E5	A9	E8	00232	BLBS	LCLSYMGSD, 29\$	2186	
		06	E6	A9	E9	00236	BLBC	WORDPSECTGSD, 27\$	2187	
		50	04	A5	3C	0023A	MOVZWL	4(R5), R0	2188	
				04	11	0023E	BRB	28\$		
		50	04	A5	9A	00240	MOVZBL	4(R5), R0	2189	
				50	DD	00244	PUSHL	R0	2187	
				04	11	00246	BRB	30\$		
	FA3C	7E	06	A5	3C	00248	MOVZWL	6(R5), -(SP)	2191	
		CA		01	FB	0024C	CALLS	#1, FNDPSCMAPENT	2186	
		52		50	DD	00251	MOVL	R0, PSCTMAPENT		
		53		62	DD	00254	MOVL	(PSCTMAPENT), PSCTDESC	2193	
		50		69	DD	00257	MOVL	SYMTABENT, R0	2197	
				53	D5	0025A	TSTL	PSCTDESC	2195	
				03	13	0025C	BEQL	31\$		
				01B3	31	0025E	BRW	51\$		
				01BB	31	00261	BRW	52\$		
		50		69	DD	00264	MOVL	SYMTABENT, R0	2208	
37	0B	A0		03	E1	00267	BBC	#3, 11(R0), 33\$		
		51	20	A0	DD	0026C	MOVL	32(R0), SYMBOLCLUSTER	2209	
				31	13	00270	BEQL	33\$		
		2D	58	A1	E9	00272	BLBC	88(SYMBOLCLUSTER), 33\$	2211	
		56	00000000G	00	DD	00276	MOVL	LNK\$GL_CURCLU, R6	2213	
21	58	A6		02	E1	0027D	BBC	#2, 88(R6), 33\$		
		56	00000000G	00	DD	00282	MOVL	LNK\$GL_CURCLU, R6	2214	
		16	58	A6	E9	00289	BLBC	88(R6), 33\$		
50		60	4C	A1	C1	0028D	ADDL3	76(SYMBOLCLUSTER), (R0), R0	2215	
		51	00000000G	00	DD	00292	MOVL	LNK\$GL_CURCLU, R1	2217	
51		54	4C	A1	C1	00299	ADDL3	76(R1), SYMBOLVALUE, R1		
		51		50	D1	0029E	CMPL	R0, R1		
				56	13	002A1	BEQL	36\$		
7E	00000000G	00		14	C1	002A3	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	2221	
			50	A9	9F	002AB	PUSHAB	OBMODESC+40-		
			FC	A9	DD	002AE	PUSHL	SYMBOLSTRING		
				03	DD	002B1	PUSHL	#3		
			00000000G	8F	DD	002B3	PUSHL	#LINS MULDEF		
	00000000G	00		05	FB	002B9	CALLS	#5, LTB\$SIGNAL		
35		67		03	E0	002C0	BBS	#3, (R7), 36\$	2223	
		50		69	DD	002C4	MOVL	SYMTABENT, R0	2224	
2D	0A	A0		03	E0	002C7	BBS	#3, 10(R0), 36\$		
	00	B9		54	DD	002CC	MOVL	SYMBOLVALUE, @SYMTABENT	2225	
				27	11	002D0	BRB	36\$	2250	
		67		01	E0	002D2	BBS	#1, (R7), 37\$	2251	
26		01		00	EF	002D6	EXTZV	#0, #1, (R7), -(SP)	2253	
67				7E	D4	002DB	CLRL	-(SP)		
		6A		02	FB	002DD	CALLS	#2, CREFILTER		
		16		67	E8	002E0	BLBS	(R7), 36\$	2255	
		50		69	DD	002E3	MOVL	SYMTABENT, R0	2258	
		0F	0A	A0	E9	002E6	BLBC	10(R0), 36\$		
		A0		01	8A	002EA	BICB2	#1, 10(R0)	2260	
		04	0B	A0	E9	002EE	BLBC	11(R0), 35\$	2262	
				6B	B6	002F2	INCW	LNK\$GW_NUDFLSYMS	2263	

				03	11	002F4	BRB	36\$		
				AB	B6	002F6	INCW	LNK\$GW_NUDFSYMS		2264
				012C	31	002F9	BRW	53\$		2266
				69	D0	002FC	MOVL	SYMTABENT, R0		2272
				60	D5	002FF	TSTL	(R0)		
				25	13	00301	BEQL	40\$		
				60	0F	00303	REMQUE	(R0), NXTSYMENT		2274
				51	D1	00306	CMPL	NXTSYMENT, LNK\$GL_LIBSYM		2276
				07	12	0030D	BNEQ	38\$		
				61	D0	0030F	MOVL	(NXTSYMENT), LNK\$GL_LIBSYM		2277
				69	D0	00316	MOVL	SYMTABENT, R0		2279
				0A	A0	E8	BLBS	10(R0), 40\$		
				0B	A0	E9	BLBC	11(R0), 39\$		2281
				04	B7	00321	DECW	LNK\$GW_NUDFLSYMS		2282
				03	11	00323	BRB	40\$		
				AB	B7	00325	DECW	LNK\$GW_NUDFSYMS		2283
				69	D0	00328	MOVL	SYMTABENT, R0		2294
				0A	A0	9E	MOVAB	10(R0), R1		
				53	3C	0032F	MOVZWL	(R1), R3		
				53	8F	CA	BICL2	#-8453, R3		
				52	67	3C	MOVZWL	(R7), R2		2296
				52	04	CA	BICL2	#4, R2		
				52	53	A9	BISW3	R3, R2, (R1)		
				0F	AC	F0	INSV	ENTMSKFLG, #15, #1, (R1)		2298
				00	02	E1	BBC	#2, LNK\$GL_CTLMSK+3, 41\$		2300
				61	04	88	BISB2	#4, (R1)		2301
				0E	A5	90	MOVB	1(R5), 14(R0)		2303
				60	54	D0	MOVL	SYMBOLVALUE, (R0)		2304
				08	A9	B0	MOVW	ENTRYMASK, 8(R0)		2305
				01	02	EF	EXTZV	#2, #1, OBMODESC+20, R0		2306
				0B	50	F0	INSV	R0, #11, #1, (R1)		
				12	A9	E8	BLBS	LCLSYMGSD, 44\$		2307
				06	A9	E9	BLBC	WORDPSECTGSD, 42\$		2308
				50	A5	3C	MOVZWL	4(R5), R0		2309
				50	04	11	BRB	43\$		
				50	A5	9A	MOVZBL	4(R5), R0		2310
				50	DD	0037E	PUSHL	R0		2308
				04	11	00380	BRB	45\$		
				7E	A5	3C	MOVZWL	6(R5), -(SP)		2312
				CA	01	FB	CALLS	#1, FNDPSCMAPENT		2307
				52	50	D0	MOVL	R0, PSCTMAPENT		
				53	62	D0	MOVL	(PSCTMAPENT), PSCTDESC		2314
				50	69	D0	MOVL	SYMTABENT, R0		2315
				20	00	D0	MOVL	LNK\$GL_CURCLU, 32(R0)		
				14	AB	3C	MOVZWL	LNK\$GW_NMODULES, 20(R0)		2316
				0B	03	E1	BBC	#3, 11(R0), 46\$		2318
				0A	04	8A	BICB2	#4, 10(R0)		2320
				12	58	E9	BLBC	FOUND, 48\$		2322
				0A	06	E0	BBS	#6, 11(R0), 47\$		
				00E6	00	FB	CALLS	#0, SHRIMGSYM		2323
				08	03	11	BRB	47\$		2318
				50	AB	B6	INCW	LNK\$GW_LSYMBOLS		2329
				50	58	E8	BLBS	FOUND, 49\$		2331
				0A	69	D0	MOVL	SYMTABENT, R0		2332
				3E	03	E0	BBS	#3, 11(R0), 50\$		
				67	00	EF	EXTZV	#0, #1, (R7), -(SP)		2334
				01	01	DD	PUSHL	#1		

LNK_OBJPASS1
V04=000

J 16
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 65
(16)

	6A		02	FB	003CE	CALLS	#2, CREFILTER	:	
	2D	00000000G	00	E9	003D1	BLBC	LNK\$GL_CTLMSK+1, 50\$	:	2336
	50		69	D0	003D8	MOVL	SYMTABENT, R0	:	2337
25	0A	A0	03	E0	003DB	BBS	#3, 10(R0), 50\$	:	
	50		69	D0	003E0	MOVL	SYMTABENT, R0	:	2338
1D	0B	A0	05	E0	003E3	BBS	#5, 11(R0), 50\$	:	
	50		7E	D4	003E8	CLRL	-(SP)	:	2342
	7E	0A	69	D0	003EA	MOVL	SYMTABENT, R0	:	2343
	7E	A9	A0	3C	003ED	MOVZWL	10(R0), -(SP)	:	
			04	C1	003F1	ADDL3	#4, SYMENTNAM, -(SP)	:	2342
			50	DD	003F6	PUSHL	R0	:	
		00000000G	00	9F	003F8	PUSHAB	LNK\$AL_VALCTLTB	:	2340
	00		05	FB	003FE	CALLS	#5, CRF\$INSRTREF	:	2342
	50		69	D0	00405	MOVL	SYMTABENT, R0	:	2351
			53	D5	00408	TSTL	PSCTDESC	:	2348
			13	13	0040A	BEQL	52\$	:	
	51	04	A2	D0	0040C	MOVL	4(PSCTMAPENT), MODPSCONTRIBLK	:	2350
	60	08	A1	C0	00410	ADDL2	8(MODPSCONTRIBLK), (R0)	:	2352
	04	14	A3	D0	00414	MOVL	20(PSCTDESC), 4(R0)	:	2353
	14	A3	50	D0	00419	MOVL	R0, 20(PSCTDESC)	:	2354
			09	11	0041D	BRB	53\$	:	2348
	04	A0	04	A2	D0	MOVL	4(PSCTMAPENT), 4(R0)	:	2357
	04	A2	50	D0	00424	MOVL	R0, 4(PSCTMAPENT)	:	2358
	50		01	D0	00428	MOVL	#1, R0	:	2361
			50	04	0042B	RET		:	2250
				D4	0042C	CLRL	R0	:	2364
				04	0042E	RET		:	

; Routine Size: 1071 bytes, Routine Base: \$CODE\$ + 0C97

; 1883 2365 1


```
1885 2366 1 routine symbols =
1886 2367 2 begin
1887 2368 2 local
1888 2369 2 length;
1889 2370 2 bind
1890 2371 2 symbolrec = objvec [.gsdoffset] : block [, byte];
1891 2372 2
1892 2373 2 entrymask = 0; ! ZERO THE ENTRY MASK SINCE NONE EXISTS
1893 2374 2
1894 2375 2 if not .symbolrec [gsy$y_def] ! IF NOT DEFINED
1895 2376 2 then begin
1896 2377 2 length = symbolrec [srf$t_name] - symbolrec [srf$t_start] + .symbolrec [srf$b_namlng];
1897 2378 2 symbolstring = symbolrec [srf$b_namlng];
1898 2379 2 end
1899 2380 2 else if .wordpsectgsd ! IF A WORD OF PSECT NUMBER
1900 2381 2 then begin
1901 2382 2 length = symbolrec [sdfw$t_name] - symbolrec [sdfw$t_start] + .symbolrec [sdfw$b_namlng];
1902 2383 2 symbolstring = symbolrec [sdfw$b_namlng];
1903 2384 2 end
1904 2385 2 else begin
1905 2386 2 length = symbolrec [sdf$t_name] - symbolrec [sdf$t_start] + .symbolrec [sdf$b_namlng];
1906 2387 2 symbolstring = symbolrec [sdf$b_namlng];
1907 2388 2 end;
1908 2389 2
1909 2390 2 if not prosymbol (0) ! GO PROCESS THIS SYMBOL
1910 2391 2 then return false; ! AND EXIT IF FAILURE
1911 2392 2
1912 2393 2 gsdoffset = .gsdoffset + .length; ! UPDATE THE GSD OFFSET FOR NEXT
1913 2394 2 return true;
1914 2395 1 end;
```

			001C 00000	SYMBOLS: .WORD	Save R2,R3,R4	2366
	54	00000000'	EF 9E 00002	MOVAB	SYMBOLSTRING, R4	
	50	F0	A4 3C 00009	MOVZWL	GSDOFFSET, R0	2371
	50	FC	A4 C0 0000D	ADDL2	OBJVEC, R0	
		F2	A4 B4 00011	CLRW	ENTRYMASK	2373
15	02	A0	01 E0 00014	BBS	#1, 2(R0), 1\$	2375
51		50	50 C3 00019	SUBL3	R0, R0, R1	2377
		52	A0 9A 0001D	MOVZBL	4(R0), R2	
		51	52 C0 00021	ADDL2	R2, R1	
		52	A1 9E 00024	MOVAB	5(R1), LENGTH	
		64	A0 9E 00028	MOVAB	4(R0), SYMBOLSTRING	2378
			2C 11 0002C	BRB	3\$	2375
	15	EA	A4 E9 0002E	1\$: BLBC	WORDPSECTGSD, 2\$	2380
51		50	50 C3 00032	SUBL3	R0, R0, R1	2382
		53	0A A0 9A 00036	MOVZBL	10(R0), R3	
		51	53 C0 0003A	ADDL2	R3, R1	
		52	0B A1 9E 0003D	MOVAB	11(R1), LENGTH	
		64	0A A0 9E 00041	MOVAB	10(R0), SYMBOLSTRING	2383
			13 11 00045	BRB	3\$	2380
51		50	50 C3 00047	2\$: SUBL3	R0, R0, R1	2386
		53	09 A0 9A 0004B	MOVZBL	9(R0), R3	
		51	53 C0 0004F	ADDL2	R3, R1	

LNK_OBJPASS1
V04=000

L 16
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 67
(17)

	52	0A	A1	9E	00052	MOVAB	10(R1), LENGTH	:	
	64	09	A0	9E	00056	MOVAB	9(R0), SYMBOLSTRING	:	2387
			7E	D4	0005A	CLRL	-(SP)	:	2390
FB70	CF		01	FB	0005C	CALLS	#1, PROSYMBOL	:	
	08		50	E9	00061	BLBC	R0, 4\$	:	
F0	A4		52	A0	00064	ADDW2	LENGTH, GSDOFFSET	:	2393
	50		01	D0	00068	MOVL	#1, R0	:	2394
				04	0006B	RET		:	
			50	D4	0006C	CLRL	R0	:	2395
				04	0006E	RET		:	

; Routine Size: 111 bytes, Routine Base: \$CODE\$ + 10C6

; 1915 2396 1


```
: 1917      2397 1 routine entpnts =
: 1918      2398 2 begin
: 1919      2399 2 local
: 1920      2400 2     length;
: 1921      2401 2 bind
: 1922      2402 2     symbolrec = objvec [.gsdoffset] : block [, byte];
: 1923      2403 2
: 1924      2404 2 if .wordpsectgsd
: 1925      2405 2 then begin
: 1926      2406 2     entrymask = .symbolrec [epmw$w_mask];           ! EXTRACT THE ENTRY POINT MASK
: 1927      2407 2     symbolstring = symbolrec [epmw$b_namlng];      ! POINT TO THE SYMBOL
: 1928      2408 2     length = symbolrec [epmw$t_name] - symbolrec [epmw$t_start] + .symbolrec [epmw$b_namlng];
: 1929      2409 2     end
: 1930      2410 2 else begin
: 1931      2411 2     entrymask = .symbolrec [epm$w_mask];           ! EXTRACT THE ENTRY POINT MASK
: 1932      2412 2     symbolstring = symbolrec [epm$b_namlng];      ! POINT TO THE SYMBOL
: 1933      2413 2     length = symbolrec [epm$t_name] - symbolrec [epm$t_start] + .symbolrec [epm$b_namlng];
: 1934      2414 2     end;
: 1935      2415 2
: 1936      2416 2 if not prosymbol (1)                               ! GO PROCESS THE SYMBOL
: 1937      2417 2 then return false;                                ! AND EXIT IF FAILURE
: 1938      2418 2
: 1939      2419 2 gsdoffset = .gsdoffset + .length;                 ! ELSE UPDATE THE OFFSET FOR NEXT
: 1940      2420 2 return true;
: 1941      2421 1 end;
```

				000C 00000	ENTPNTS: .WORD	Save R2,R3		2397
	53	00000000'	EF	9E 00002	MOVAB	GSDOFFSET, R3		
	52		63	3C 00009	MOVZWL	GSDOFFSET, R2		2402
	52	0C	A3	C0 0000C	ADDL2	OBJVEC, R2		
	1B	FA	A3	E9 00010	BLBC	WORDPSECTGSD, 1\$		2404
	02	A3	A2	B0 00014	MOVW	10(R2), ENTRYMASK		2406
	10	A3	A2	9E 00019	MOVAB	12(R2), SYMBOLSTRING		2407
50	52	0C	52	C3 0001E	SUBL3	R2, R2, R0		2408
	51	0C	A2	9A 00022	MOVZBL	12(R2), R1		
	50		51	C0 00026	ADDL2	R1, R0		
	52	0D	A0	9E 00029	MOVAB	13(R0), LENGTH		
			19	11 0002D	BRB	2\$		2404
	02	A3	09	A2 B0 0002F	MOVW	9(R2), ENTRYMASK		2411
	10	A3	0B	A2 9E 00034	MOVAB	11(R2), SYMBOLSTRING		2412
50	52	0B	52	C3 00039	SUBL3	R2, R2, R0		2413
	51	0B	A2	9A 0003D	MOVZBL	11(R2), R1		
	50		51	C0 00041	ADDL2	R1, R0		
	52	0C	A0	9E 00044	MOVAB	12(R0), LENGTH		
			01	DD 00048	PUSHL	#1		2416
FB13	CF		01	FB 0004A	CALLS	#1, PROSYMBOL		
	07		50	E9 0004F	BLBC	R0, 3\$		
	63		52	A0 00052	ADDW2	LENGTH, GSDOFFSET		2419
	50		01	D0 00055	MOVL	#1, R0		2420
				04 00058	RET			
			50	D4 00059	CLRL	R0		2421
				04 0005B	RET			

LNK_OBJPASS1
V04=000

B 1
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 69
(18)

; Routine Size: 92 bytes, Routine Base: \$CODE\$ + 1135


```
: 1943      2422 1 routine procedef =
: 1944      2423 2   begin
: 1945      2424 2
: 1946      2425 2       A PROCEDURE DEFINITION IS AN EXTENDED ENTRY POINT DEFINITION, CARRYING WITH
: 1947      2426 2       IT A DESCRIPTION OF THE PROCEDURE'S FORMAL ARGUMENTS. PROCESSING THESE CONSISTS
: 1948      2427 2       IN NORMAL SYMBOL DEFINITION PROCESSING FOLLOWED BY:-
: 1949      2428 2           (1) VALIDATION OF THE FORMAT OF FORMAL DESCRIPTION (I.E. JUST CHECK
: 1950      2429 2               THAT MINIMUM NUMBER OF ARGUMENTS SPECIFIED IS LESS THAN
: 1951      2430 2               OR EQUAL TO THE MAXIMUM.
: 1952      2431 2           (2) ALLOCATION OF AN ARGUMENT VALIDATION DATA ARRAY WHICH
: 1953      2432 2               HAS THE FOLLOWING FORMAT:
: 1954      2433 2               -----
: 1955      2434 2               ! N+2      ! LENGTH OF ARRAY WHERE N = MAXIMUM # ARGUMENTS
: 1956      2435 2               -----
: 1957      2436 2               ! MIN      ! MINIMUM NUMBER OF ARGS
: 1958      2437 2               -----
: 1959      2438 2               ! PASS M 1 ! PASSING MECHANISM OF ARG 1
: 1960      2439 2               -----
: 1961      2440 2               ! PASS M 2 !      "      "      "      2
: 1962      2441 2               -----
: 1963      2442 2               ETC TO N ARGUMENTS
: 1964      2443 2
: 1965      2444 2       NOTE THAT ONLY THE VALIDATION CONTROL BYTE, ARGUMENT PASSING MECHANISM
: 1966      2445 2       IS EXTRACTED FROM THE FORMAL ARGUMENT DESCRIPTORS. THE REST OF SUCH
: 1967      2446 2       DESCRIPTORS IS IGNORED.
: 1968      2447 2
: 1969      2448 2       THE SYMBOL TABLE ENTRY POINTS TO THIS ARRAY. NOTE THAT MULTIPLE DEFINITION OF THE
: 1970      2449 2       SAME PROCEDURE WILL REPLACE THE VALIDATION DATA WITH NEW DATA.
: 1971      2450 2
: 1972      2451 2 local
: 1973      2452 2     entflag,
: 1974      2453 2     argvaldata : ref vector [, byte],
: 1975      2454 2     argcount;
: 1976      2455 2
: 1977      2456 2 if not entpnts () then return false;      ! PROCESS THE ENTRY POINT SYMBOL PART
: 1978      2457 2
: 1979      2458 2 entflag = .symtabent neq 0;      ! SEE IF ENTERED
: 1980      2459 2 begin
: 1981      2460 2
: 1982      2461 2 bind
: 1983      2462 2   formals = objvec [.gsdoffset] : block [, byte]; ! THE FORMAL DESCRIPTION
: 1984      2463 2
: 1985      2464 2 if .formals [fml$b_minargs] gtru .formals [fml$b_maxargs] ! IF THERE IS AN INCONSISTENT
: 1986      2465 2 then
: 1987      2466 2   begin
: 1988      2467 2     signal (lin$_illfmlcnt, 5,      ! DESCRIPTION OF FORMALS
: 1989      2468 2     .formals [fml$b_minargs], .formals [fml$b_maxargs],      ! OUTPUT ERROR MESSAGE
: 1990      2469 2     (if .wordpsectgsd then objrec [pro$b_namlng] else objrec [pro$b_namlng]),
: 1991      2470 2     obmodesc [omd$b_namlng], lnk$gl_curfi [fdb$b_filename]);
: 1992      2471 2   return false;
: 1993      2472 2   end;
: 1994      2473 2
: 1995      2474 2 gsdoffset = .gsdoffset + fml$c_size;      ! UPDATE RECORD POINTER
: 1996      2475 2
: 1997      2476 2 if .entflag and (argvaldata = .symtabent [sym$l_valdata]) neq 0 ! IF VALIDATION DATA ALREADY EXISTS
: 1998      2477 2 then
: 1999      2478 2   begin
:           ! MUST BE A NEW DEFINITION
```



```
: 2000      2479  4      lnk$dealblk (.argvaldata [0], argvaldata [0]); ! SO DISPOSE OF OLD DATA
: 2001      2480  4      symtabent [sym$l_valdata] = 0; ! AND FORGET IT
: 2002      2481  4      end;
: 2003      2482  4
: 2004      2483  4      if (argcount = .formals [fml$b_maxargs]) eql 0 ! IF MAXIMUM NUMBER ARGUMENTS
: 2005      2484  4      then
: 2006      2485  4          return true; ! IS ZERO, ALL DONE
: 2007      2486  4
: 2008      2487  4      if .entflag
: 2009      2488  4      then
: 2010      2489  4          begin
: 2011      2490  4              lnk$alloblk (.argcount + 2, symtabent [sym$l_valdata]); ! ALLOCATE A VALIDATION DATA BLOCK
: 2012      2491  4              argvaldata = .symtabent [sym$l_valdata];
: 2013      2492  4              argvaldata [0] = .argcount + 2; ! SET ITS SIZE
: 2014      2493  4              argvaldata [1] = .formals [fml$b_minargs]; ! SAVE MINIMUM ARGUMENT SPECIFICATION
: 2015      2494  4              end;
: 2016      2495  4
: 2017      2496  4      incr i from 1 to .argcount ! BEGIN A LOOP THROUGH
: 2018      2497  3      do
: 2019      2498  4          begin ! ALL FORMAL ARGUMENT DESCRIPTORS
: 2020      2499  4
: 2021      2500  4          bind
: 2022      2501  4              argdesc = objvec [.gsdoffset] : block [, byte]; ! POINT TO NEXT DESCRIPTOR
: 2023      2502  4
: 2024      2503  4          if .entflag then argvaldata [.i + 1] = .argdesc [arg$v_passmech];
: 2025      2504  4
: 2026      2505  4              ! EXTRACT THE PASSING MECHANISM
: 2027      2506  4          gsdoffset = .gsdoffset + .argdesc [arg$b_bytecnt] + ! AND UPDATE RECORD POINTER
: 2028      2507  4          arg$c_size;
: 2029      2508  3      end;
: 2030      2509  3
: 2031      2510  3      return true;
: 2032      2511  2      end;
: 2033      2512  1      end;
```

```
007C 00000 PROCEDEF:
          56 00000000' EF 9E 00002 .WORD Save R2,R3,R4,R5,R6 : 2422
97 AF 00000000 00 FB 00009 MOVAB SYMTABENT, R6 : 2456
4C 50 E9 0000D CALLS #0, ENTPNTS : 2458
50 D4 00010 BLBC R0, 4$
66 D5 00012 CLRL R0
02 13 00014 TSTL SYMTABENT
50 D6 00016 BEQL 1$
55 50 D0 00018 1$: INCL R0
51 F8 A6 D0 0001B MOVL R0, ENTFLAG : 2462
54 51 EC A6 3C 0001F MOVL OBJVEC, R0
01 50 51 C1 00023 MOVZWL GSDOFFSET, R1 : 2464
A4 64 91 00027 ADDL3 R1, R0, R4
7E 00000000G 00 31 1B 0002B CMPB (R4), 1(R4) : 2470
50 14 C1 0002D BLEQU 5$
05 E6 A6 E9 00035 ADDL3 #20, LNK$GL_CURFIL, -(SP) : 2469
E6 A6 E9 00038 BLBC OBMODESC+40-
WORDPSECTGSD, 2$
```


	50		0C	C0	0003C	ADDL2	#12, R0		
			03	11	0003F	BRB	3\$		
	50		0B	C0	00041	ADDL2	#11, R0		
			50	DD	00044	PUSHL	R0		
	7E	01	A4	9A	00046	MOVZBL	1(R4), -(SP)		2470
	7E		64	9A	0004A	MOVZBL	(R4), -(SP)		
			05	DD	0004D	PUSHL	#5		
00000000G	00	00000000G	8F	DD	0004F	PUSHL	#LINS ILLFMLCNT		
			07	FB	00055	CALLS	#7, LIB\$SIGNAL		
			7C	11	0005C	BRB	12\$		2471
EC	A6		02	A0	0005E	ADDW2	#2, GSDOFFSET		2474
	1B		55	E9	00062	BLBC	ENTFLAG, 6\$		2476
	50		66	D0	00065	MOVL	SYMTABENT, R0		
	53	18	A0	D0	00068	MOVL	24(R0), ARGVALDATA		
			12	13	0006C	BEQL	6\$		
			53	DD	0006E	PUSHL	ARGVALDATA		2479
	7E		63	9A	00070	MOVZBL	(ARGVALDATA), -(SP)		
00000000G	00		02	FB	00073	CALLS	#2, LNK\$DEALBLK		
	50		66	D0	0007A	MOVL	SYMTABENT, R0		2480
		18	A0	D4	0007D	CLRL	24(R0)		
	52	01	A4	9A	00080	MOVZBL	1(R4), ARGCOUNT		2483
			50	13	00084	BEQL	11\$		
	20		55	E9	00086	BLBC	ENTFLAG, 7\$		2487
7E	66		18	C1	00089	ADDL3	#24, SYMTABENT, -(SP)		2490
		02	A2	9F	0008D	PUSHAB	2(ARGCOUNT)		
00000000G	00		02	FB	00090	CALLS	#2, LNK\$ALLOBLK		
	50		66	D0	00097	MOVL	SYMTABENT, R0		2491
	53	18	A0	D0	0009A	MOVL	24(R0), ARGVALDATA		
	50	02	A2	9E	0009E	MOVAB	2(ARGCOUNT), R0		2492
	63		50	90	000A2	MOVB	R0, (ARGVALDATA)		
01	A3		64	9C	000A5	MOVB	(R4), 1(ARGVALDATA)		2493
			51	D4	000A9	CLRL	I		2501
			25	11	000AB	BRB	10\$		
	50	EC	A6	3C	000AD	MOVZWL	GSDOFFSET, R0		
	50	F8	A6	C0	000B1	ADDL2	OBJVEC, R0		
54	0A		55	E9	000B5	BLBC	ENTFLAG, 9\$		2503
	02		00	EF	000B8	EXTZV	#0, #2, (R0), R4		
	01	A143	54	90	000BD	MOVB	R4, 1(I)[ARGVALDATA]		
			54	A6	3C	MOVZWL	GSDOFFSET, R4		2506
	50	EC	A0	9A	000C6	MOVZBL	1(R0), R0		
	50	01	54	C0	000CA	ADDL2	R4, R0		
EC	A6		02	A1	000CD	ADDW3	#2, R0, GSDOFFSET		
D7			52	F3	000D2	AOBLEQ	ARGCOUNT, I, 8\$		2496
			51	D0	000D6	MOVL	#1, R0		2510
			50	04	000D9	RET			
						CLRL	R0		2512
			50	D4	000DA	RET			
			04	000DC					

; Routine Size: 221 bytes, Routine Base: \$CODE\$ + 1191


```
: 2035      2513 1 routine proeom (seqchkflg) =
: 2036      2514 2   begin
: 2037      2515 2   :
: 2038      2516 2       PROCESS END OF MODULE RECORDS:
: 2039      2517 2       (1) VALIDATE SEQUENCE
: 2040      2518 2       (2) INTERPRET COMPILER COMPLETION CODE,
: 2041      2519 2       ISSUING APPROPRIATE ERROR OR WARNING
: 2042      2520 2       MESSAGE
: 2043      2521 2       (3) VALIDATE THE TRANSFER ADDRESS
: 2044      2522 2       (IF ANY) FOR:
: 2045      2523 2       (A) P-SECTION DEFINED
: 2046      2524 2       (B) P-SECTION EXECUTABLE AND RELOCATABLE
: 2047      2525 2       (C) ADDRESS IS WITHIN THAT P-SECTION
: 2048      2526 2       (D) NOT A MULTIPLE SPECIFICATION -( RETAIN THE FIRST)
: 2049      2527 2       (4) RECORD THE TRANSFER ADDRESS, SEPARATING AN ADDRESS IN
: 2050      2528 2       THE DEBUGGER FROM THAT OF THE USER.
: 2051      2529 2       (5) TRUNCATE THE MODULE DESCRIPTOR (MAPPING TABLE PORTION) TO ITS
: 2052      2530 2       CORRECT SIZE FOR NUMBER OF P-SECTS CONTRIBUTED TO.
: 2053      2531 2   :
: 2054      2532 2   bind
: 2055      2533 2       eomrec = .objvec : block [, byte];
: 2056      2534 2   builtin
: 2057      2535 2       nullparameter;
: 2058      2536 2   local
: 2059      2537 2       comcode,
: 2060      2538 2       deferredsym : ref block [, byte],      ! POINTER TO DEFERRED SYMBOL
: 2061      2539 2       deferredsnb : ref block [, byte],      ! DEFERRED SYMBOL NAME BLOCK
: 2062      2540 2       modpscontriblk : ref block [, byte],    ! THE MODULE'S CONTRIBUTION TO P-SECT
: 2063      2541 2       wordpsecteom,                          ! TRUE IF EOMW RATHER THAN EOM
: 2064      2542 2       psctmap : ref block [, byte],          ! PSECT MAPPING TABLE POINTER
: 2065      2543 2       psctnum,                                ! P-SECTION NUMBER OF TRANSFER ADDRESS
: 2066      2544 2       tfrweak,                                ! TRUE IF TRANSFER ADDRESS THIS EOM IS WEAK
: 2067      2545 2       tfradr,                                ! TEMP FOR TRANSFER ADDRESS
: 2068      2546 2       tfrpsc : ref block [, byte],           ! TEMP FOR P-SECT DESC. ADDR.
: 2069      2547 2       objdesc : ref block [, byte],         ! POINTER TO ALLOCATED OBJ MOD DESCRIPTOR
: 2070      2548 2       omdnode,
: 2071      2549 2       curcontriblk : ref block [, byte],
: 2072      2550 2       nextcontriblk : ref block [, byte],
: 2073      2551 2       objdescsize;                          ! SIZE OF OBJ MOD DESCRIPTOR TO ALLOCATE
: 2074      2552 2
: 2075      2553 2   if nullparameter (1)
: 2076      2554 2   then
: 2077      2555 2
: 2078      2556 2       if not seqchk () then return false;    ! VALIDATE THE SEQUENCE OF EOM
: 2079      2557 2
: 2080      2558 2   lnk$gl_curfil [fdb$y_omdnobin] =              ! IF ANY OBMODS FOR THIS FDB ARE
: 2081      2559 2   .lnk$gl_curfil [fdb$y_omdnobin] or .obmodesc [omd$y_nobin]; ! WITHOUT TIR RECS, FLAG IN FDB
: 2082      2560 2   wordpsecteom = (.eomrec [eom$b_rectyp] eql obj$eomw); ! DECIDE IF EOM OR EOMW
: 2083      2561 2
: 2084      2562 2   if (if .wordpsecteom                          ! CHECK LENGTH OF RECORD
: 2085      2563 3       then (.reclng neg eomw$eommin and ((.reclng lss eomw$eommx1) or (.reclng gtr eomw$eommax))) el
: 2086      2564 4       (.reclng neg eom$eommin
: 2087      2565 3           and ((.reclng lss eom$eommx1) or (.reclng gtr eom$eommax))))
: 2088      2566 2   then
: 2089      2567 3       begin
: 2090      2568 3       signal (lin$_illreclen, 3,                ! AND ISSUE ERROR
: 2091      2569 3       .reclng, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
```



```
2092 2570 3      return false;
2093 2571 3      end;
2094 2572 2
2095 2573 2      if (comcode = .eomrec [eom$b_comcod]) neq 0 ! IF NON ZERO COMPILATION CPLETE CODE
2096 2574 2      then
2097 2575 2          begin
2098 2576 2              ! CHECK
2099 2577 2
2100 2578 2          if .comcode gtru 3
2101 2579 2          then
2102 2580 2              begin
2103 2581 2                  signal (lin$_badccc, 3, ! IF AN ILLEGAL COMPLETION
2104 2582 2                      .comcode, obmodesc [omd$b_namlng], ! THEN TELL THAT AND QUIT
2105 2583 2                      lnk$gl_curfil [fdb$q_filename]);
2106 2584 2              return false;
2107 2585 2          end;
2108 2586 2
2109 2587 2      case .comcode from eom$c_warning to eom$c_abort ! SIGNAL MESSAGE BASED ON ERROR CLASS
2110 2588 2      of
2111 2589 2          set
2112 2590 2              [eom$c_warning] :
2113 2591 2                  signal (lin$_wners, 2, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
2114 2592 2
2115 2593 2              [eom$c_error] :
2116 2594 2                  begin
2117 2595 2                      signal (lin$_errors, 2, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename],
2118 2596 2                          lin$_noimgfil);
2119 2597 2                      lnk$gl_cflmsk [lnk$v_image] = false; ! DISABLE IMAGE IF ERROR
2120 2598 2                  end;
2121 2599 2
2122 2600 2              [eom$c_abort] :
2123 2601 2                  begin
2124 2602 2                      signal (lin$_eomftl, 2, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
2125 2603 2                      lnk$exit (lin$_fatalerror); ! LINK ABORT
2126 2604 2                  end;
2127 2605 2          tes;
2128 2606 2
2129 2607 2      end;
2130 2608 2
2131 2609 2      if .obmodesc [omd$w_hipsct] eql lnk$c_maxpsects ! IF HIGHEST NUMBERED P-SECT
2132 2610 2          and .obmodesc [omd$v_nopsct] ! IS LNK$c_MAXPSECTS AND STILL NO DEFINITIONS
2133 2611 2      then ! BEEN SEEN
2134 2612 2          signal_stop (lin$_nopscts, 2, ! THERE IS A FORMAT ERROR
2135 2613 2              obmodesc [omd$b_namlng], ! ISSUE MESSAGE
2136 2614 2              lnk$gl_curfil [fdb$q_filename], lin$_format);
2137 2615 2
2138 2616 2      if (if .wordpsecteom then (.reclng geq eomw$c_eommx1) else (.reclng geq eom$c_eommx1))
2139 2617 2          ! IF RECORD LENGTH IS (MX1)
2140 2618 2          ! OR (MAX)
2141 2619 2      then
2142 2620 2          begin ! A TRANSFER ADDRESS EXISTS, SO
2143 2621 2              psctnum = (if .wordpsecteom ! GET THE P-SECTION INDEX AND
2144 2622 2                  then .eomrec [eomw$w_psindx] else .eomrec [eom$b_psindx]);
2145 2623 2
2146 2624 2              if .psctnum gtru .obmodesc [omd$w_hipsct] or .obmodesc [omd$v_nopsct]
2147 2625 2          then
2148 2626 2              begin
```



```
2149 2627 4      signal (lin$_badpsc, 3,      ! IF P-SECTION NOT DEFINED
2150 2628 4      .psctnum, obmodesc [omd$b_namlng],      ! THEN OUTPUT ERROR AND QUIT
2151 2629 4      lnk$gl_curfil [fdb$q_filename]);
2152 2630 4      return false;
2153 2631 4      end
2154 2632 3  else
2155 2633 4      begin
2156 2634 4      psctmap = findpscmappent (.psctnum); ! GET MAPPING TABLE ENTRY
2157 2635 4
2158 2636 4      if (tfrpsc = .psctmap [pmt$l_pscdes]) eql 0 ! SET THE POINTER TO THAT P-SECTION
2159 2637 4      then
2160 2638 5      begin
2161 2639 5      signal (lin$_badpsc, 3,      ! BUT ISSUE AN ERROR IF UNDEFINED
2162 2640 5      .psctnum, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
2163 2641 5      return false;
2164 2642 4      end;
2165 2643 4
2166 2644 4      if not .tfrpsc [psc$v_exe]      ! IF THE P-SECTION CONTAINING THE
2167 2645 4      or not .tfrpsc [psc$v_rel]      ! TRANSFER ADDRESS IS NOT
2168 2646 4      then
2169 2647 4      signal (lin$_pscnxr, 2,      ! THEN ISSUE AN ERROR MESSAGE
2170 2648 4      obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
2171 2649 4
2172 2650 4      modpscontriblk = .psctmap [pmt$l_modcon];
2173 2651 4      tfradr = (if .wordpsecteom then .eomrec [eomw$l_tfradr] else .eomrec [eom$l_tfradr]) +
2174 2652 4      ! BUT IN ANY CASE COMPUTE THE
2175 2653 4      .modpscontriblk [mpc$l_offset];      ! ADDRESS RELATIVE TO THE WHOLE P-SECT
2176 2654 6      tfrweak = (if .wordpsecteom then ((.reclng eql eomw$c_eommax) and
2177 2655 6      ! SET FLAG FOR WEAKNESS OF TRANSFER ADDRESS
2178 2656 4      .eomrec [eomw$v_wktfr]) else ((.reclng eql eom$c_eommax) and .eomrec [eom$v_wktfr]));
2179 2657 4
2180 2658 4      if not .lnk$gl_curfil [fdb$v_debugger]      ! IF THE CURRENT INPUT FILE
2181 2659 4      and (.lnk$gl_tfrpsc eql 0      ! DOES NOT CONTAIN THE DEBUGGER
2182 2660 5      or (.weaktfradr and not .tfrweak))      ! AND NO PREVIOUS TRANSFER ADDRESS WAS SPECIFIED
2183 2661 5      ! OR IT WAS A WEAK TRANSFER ADDRESS AND THIS ONE IS NOT
2184 2662 5      then
2185 2663 4      begin
2186 2664 5      if (not .lnk$gl_ctlmsk [lnk$v_imgidopt]) and      ! IF IMAGE ID NOT MANUALLY SET
2187 2665 5      (.lnk$gt_imgid [0] eql 0      ! IF STILL NO IMAGE IDENT
2188 2666 6      or .lnk$gl_tfrpsc eql 0)      ! OR IDENT WAS FROM MODULE W/O TRANSFER ADDRESS
2189 2667 6      then
2190 2668 6      ch$move (.imageidstring [0] + 1, imageidstring [0],      ! THEN MOVE IN THE ONE OF
2191 2669 5      lnk$gt_imgid [0]);      ! THIS MODULE AS SAVED ON MHD RECORD
2192 2670 5
2193 2671 5      lnk$gl_tfrpsc = .tfrpsc;      ! THEN THIS IS THE USER'S
2194 2672 5      lnk$gl_tfradr = .tfradr;      ! TRANSFER ADDRESS
2195 2673 5      weaktfradr = .tfrweak;      ! REMEMBER IF THIS WAS WEAK OR STRONG TRANSFER ADDRESS
2196 2674 5      end
2197 2675 5  else
2198 2676 5
2199 2677 4      if .lnk$gl_curfil [fdb$v_debugger]
2200 2678 4      and .lnk$gl_dbgtfps eql 0      ! OTHERWISE IF THIS IS THE DEBUGGER FILE AND THERE HAS NOT
2201 2679 4      ! PREVIOUSLY BEEN A DEBUGGER TRANSFER
2202 2680 4      then
2203 2681 4      begin
2204 2682 4      ! ADDRESS, MAKE THIS THE
2205 2683 5
```



```
2206      2684 5      lnk$gl_dbgtrfs = .tfrpsc;      ! DEBUGGER TRANSFER ADDRESS
2207      2685 5      lnk$gl_dbgtrf = .tfradr;
2208      2686 5      end
2209      2687 4      else
2210      2688 4
2211      2689 4          if not .tfrweak
2212      2690 4
2213      2691 4      ! THERE WAS A MULTIPLE TRANSFER ADDRESS AND THE TRANSFER ADDRESS FLAG
2214      2692 4      ! BYTE WAS NOT PRESENT, OR WAS PRESENT AND THE WEAK TRANSFER ADDRESS
2215      2693 4      ! FLAG WAS NOT ON.
2216      2694 4
2217      2695 4          then
2218      2696 4              signal (lin$_multfr, 2, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
2219      2697 4
2220      2698 3          end;
2221      2699 3
2222      2700 2      end;
2223      2701 2      ! END OF TRANSFER ADDRESS PROCESSING
2224      2702 2      if .lnk$gt_imgid [0] eql 0
2225      2703 2      then
2226      2704 2          ch$move (.imageidstring [0] + 1, imageidstring [0],
2227      2705 2          ! THEN SET THIS MODULE'S AS THE IDENT (WILL GET OVERWRITTEN IF XFR ADR
2228      2706 2          lnk$gt_imgid [0]);
2229      2707 2          ! LATER)
2230      2708 2
2231      2709 4      objdescsize = obmodesc [omd$t_pscmap] - obmodesc + ! COMPUTE SIZE OF OBJ MOD DESC TO ALLOCATE
2232      2710 4      pmt$size*((if not .obmodesc [omd$w_p256]
2233      2711 2          then .obmodesc [omd$w_hipsct] + 1
2234      2712 2          else (.obmodesc [omd$w_hipsct] + 256)/256));
2235      2713 2      lnk$alloblk (.objdescsize, objdesc);
2236      2714 2      lastobjmod [omd$l_nxtomd] = .objdesc;
2237      2715 2      ! ALLOCATE THE DESCRIPTOR
2238      2716 2      lastobjmod = .objdesc;
2239      2717 2      ! LINK INTO THE LIST
2240      2718 2      ch$move (.objdescsize, obmodesc, .objdesc);
2241      2719 2      ! AND NEW BECOMES THE LAST THEN
2242      2720 2      lib$insert tree (lnk$gl_omdtree, .objdesc [omd$l_omdnum], ! COPY THE DESCRIPTOR OUT
2243      2721 2      %ref (0), lnk$compare_omd, alloc_omdnode, omdnode, .objdesc);
2244      2722 2      ! INSERT OMD INTO OMD TREE
2245      2723 2      objdesc [omd$l_nxtomd] = 0;
2246      2724 2      ! ZERO LINK IN THE DESCRIPTOR
2247      2725 2
2248      2726 2      if .obmodesc [omd$l_lsterr] eql obmodesc [omd$l_errtxt]
2249      2727 2      then
2250      2728 2          ! CHECK FOR NO ERRORS
2251      2729 2          objdesc [omd$l_lsterr] = objdesc [omd$l_errtxt];
2252      2730 2
2253      2731 2      IF DEBUG RECORDS IN THIS OBJ MOD, THEN ADD COUNT OF PSECTS IN DEBUG OBJ MODS.
2254      2732 2      THEN RESET PER-OBJ-MOD PSECT COUNT, AND DEBUG FLAG.
2255      2733 2
2256      2734 2      if .omd_has_dbg
2257      2735 2      then
2258      2736 2          begin
2259      2737 2              lnk$gw_pscdst = .lnk$gw_pscdst + .obmodesc [omd$w_hipsct] + 1;
2260      2738 2              omd_has_dbg = false;
2261      2739 2          end;
2262      2740 2
2263      2741 2      ! FILL IN THE FIELD MPC$L_OWNO MD FOR ALL PSECTS DEFINED IN THIS MODULE
2264      2742 2      curcontriblk = .obmodesc [omd$l_nxtomd];
2265      2743 2      ! GET LIST POINTER OF PSECT CONTRIBUTION BLOCKS
```



```
: 2263      2741 2      while .curcontriblk neq 0 do
: 2264      2742      begin
: 2265      2743      nextcontriblk = .curcontriblk [mpc$l_ownomd];      ! GET LINK TO NEXT
: 2266      2744      curcontriblk [mpc$l_ownomd] = .objdesc;      ! SET OWNER POINTER
: 2267      2745      curcontriblk = .nextcontriblk;
: 2268      2746      end;
: 2269      2747
: 2270      2748      ch$fill (0, .objdescsize, obmodesc);      ! ZERO THE PROTOTYPE DESCRIPTOR
: 2271      2749      obmodesc [omd$l_lsterr] = obmodesc [omd$l_errtxt];
: 2272      2750      maxreclng = obj$c_maxrecsiz;      ! RESET TO MAX ALLOWED BY LANGUAGE
: 2273      2751      return true;
: 2274      2752      end;      ! END OF EOM PROCESSING
```

				OFFC 00000	PROEOM:	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	: 2513
				5B 00000000G	00 9E 00002	MOVAB	LNK\$GL_CURFIL, R11	
				5A 00000000'	EF 9E 00009	MOVAB	LNK\$GL_TFRPSC, R10	
				59 00000000'	EF 9E 00010	MOVAB	OBMODESC+40, R9	
				5E	0C C2 00017	SUBL2	#12, SP	
				52	A8 A9 D0 0001A	MOVL	OBJVEC, R2	: 2533
					6C 95 0001E	TSTB	(AP)	: 2553
					05 13 00020	BEQL	1\$	
				04	AC D5 00022	TSTL	4(AP)	
					0B 12 00025	BNEQ	2\$	
				F04C	CF 00 FB 00027	CALLS	#0, SEQCHK	: 2556
				03	50 E8 0002C	BLBS	R0, 2\$	
					02F0 31 0002F	BRW	46\$	
				50	6B D0 00032	MOVL	LNK\$GL_CURFIL, R0	: 2558
				01	02 EF 00035	EXTZV	#2, #1, 11(R0), R1	: 2559
				01	01 EF 0003B	EXTZV	#1, #1, OBMODESC+20, R3	
				51	53 88 00041	BISB2	R3, R1	
				02	51 F0 00044	INSV	R1, #2, #1, 11(R0)	
					51 D4 0004A	CLRL	R1	: 2560
				07	62 91 0004C	CMPB	(R2), #7	
					02 12 0004F	BNEQ	3\$	
				55	51 D6 00051	INCL	R1	
				51	51 D0 00053	MOVL	R1, WORDPSECTEOM	: 2563
				51	A9 3C 00056	MOVZWL	RECLNG, R1	: 2562
				0F	55 E9 0005A	BLBC	WORDPSECTEOM, 4\$	: 2563
				02	51 D1 0005D	CMPL	R1, #2	
					2A 13 00060	BEQL	7\$	
				08	51 B1 00062	CMPW	R1, #8	
					14 1F 00065	BLSSU	6\$	
				09	51 B1 00067	CMPW	R1, #9	
					0D 11 0006A	BRB	5\$	
				02	51 B1 0006C	CMPW	R1, #2	: 2564
					1B 13 0006F	BEQL	7\$	
				07	51 B1 00071	CMPW	R1, #7	: 2565
					05 1F 00074	BLSSU	6\$	
				08	51 B1 00076	CMPW	R1, #8	
					11 1B 00079	BLEQU	7\$	
				14	A0 9F 0007B	PUSHAB	20(R0)	: 2569
				0202	8F BB 0007E	PUSHR	#^M<R1,R9>	
					03 DD 00082	PUSHL	#3	

		00000000G	8F	DD	00084		PUSHL	#LINS_ILLRECLN		
			1B	11	0008A		BRB	8\$		
	53	01	A2	9A	0008C	7\$:	MOVZBL	1(R2), COMCODE		2573
			7D	13	00090		BEQL	14\$		
	03		53	D1	00092		CMPL	COMCODE, #3		2577
			13	1B	00095		BLEQU	9\$		
7E		6B	14	C1	00097		ADDL3	#20, LNK\$GL_CURFIL, -(SP)		2582
		0208	8F	BB	0009B		PUSHR	#M<R3,R9>		
			03	DD	0009F		PUSHL	#3		
		00000000G	8F	DD	000A1		PUSHL	#LINS_BADCCC		
			00D6	31	000A7	8\$:	BRW	22\$		
50		6B	14	C1	000AA	9\$:	ADDL3	#20, LNK\$GL_CURFIL, R0		2591
02		01	53	CF	000AE		CASEL	COMCODE, #1, #2		2586
003D		001B	0006		000B2	10\$:	.WORD	11\$-10\$,-		
								12\$-10\$,-		
								13\$-10\$		
			50	DD	000B8	11\$:	PUSHL	R0		2591
			59	DD	000BA		PUSHL	R9		
			02	DD	000BC		PUSHL	#2		
		00000000G	8F	DD	000BE		PUSHL	#LINS_WNERS		
			04	FB	000C4		CALLS	#4, LIB\$SIGNAL		
			42	11	000CB		BRB	14\$		
		00000000G	8F	DD	000CD	12\$:	PUSHL	#LINS_NOIMGFIL		2595
			50	DD	000D3		PUSHL	R0		
			59	DD	000D5		PUSHL	R9		
			02	DD	000D7		PUSHL	#2		
		00000000G	8F	DD	000D9		PUSHL	#LINS_ERRORS		
00000000G	00		05	FB	000DF		CALLS	#5, LIB\$SIGNAL		
00000000G	00		01	8A	000E6		BICB2	#1, LNK\$GL_CTLMSK		2597
			20	11	000ED		BRB	14\$		2586
			50	DD	000EF	13\$:	PUSHL	R0		2602
			59	DD	000F1		PUSHL	R9		
			02	DD	000F3		PUSHL	#2		
		00000000G	8F	DD	000F5		PUSHL	#LINS_EOMFTL		
00000000G	00		04	FB	000FB		CALLS	#4, LIB\$SIGNAL		
		00000000G	8F	DD	00102		PUSHL	#LINS_FATALERROR		2603
00000000G	00		01	FB	00108		CALLS	#1, LNK\$EXIT		
FFFF	8F	EA	A9	B1	0010F	14\$:	CMPL	OBMODESC+18, #65535		2609
			1F	12	00115		BNEQ	15\$		
	1B	EC	A9	E9	00117		BLBC	OBMODESC+20, 15\$		2610
		00000000G	8F	DD	0011B		PUSHL	#LINS_FORMAT		2614
7E		6B	14	C1	00121		ADDL3	#20, LNK\$GL_CURFIL, -(SP)		
			59	DD	00125		PUSHL	R9		2613
			02	DD	00127		PUSHL	#2		2614
		00000000G	8F	DD	00129		PUSHL	#LINS_NOPSCCTS		
00000000G	00		05	FB	0012F		CALLS	#5, LIB\$STOP		
	06		55	E9	00136	15\$:	BLBC	WORDPSECTEOM, 16\$		2616
	08	A4	A9	B1	00139		CMPL	RECLNG, #8		
			04	11	0013D		BRB	17\$		
	07	A4	A9	B1	0013F	16\$:	CMPL	RECLNG, #7		
			03	1E	00143	17\$:	BGEQU	18\$		
			0111	31	00145		BRW	38\$		
	06		55	E9	00148	18\$:	BLBC	WORDPSECTEOM, 19\$		2621
	53	02	A2	3C	0014B		MOVZWL	2(R2), PSCTNUM		2622
			04	11	0014F		BRB	20\$		
	53	02	A2	9A	00151	19\$:	MOVZBL	2(R2), PSCTNUM		
53	EA	A9	10	00	ED	20\$:	CMPL	#0, #16, OBMODESC+18, PSCTNUM		2624

					13	1F	0015B	BLSSU	21\$		
		OF	EC		A9	E8	0015D	BLBS	OBMODESC+20, 21\$		
					53	DD	00161	PUSHL	PSCTNUM		2634
	F1D4	CF			01	FB	00163	CALLS	#1, FNDPSCMAPENT		
		54			50	D0	00168	MOVL	R0, PSCTMAP		
		56			64	D0	0016B	MOVL	(PSCTMAP), TFRPSC		2636
					1A	12	0016E	BNEQ	23\$		
7E		6B			14	C1	00170	ADDL3	#20, LNK\$GL_CURFIL, -(SP)		2640
			0208		8F	BB	00174	PUSHR	#*M<R3,R9>		
					03	DD	00178	PUSHL	#3		
			00000000G		8F	DD	0017A	PUSHL	#LINS BADPSC		
	00000000G	00			05	FB	00180	CALLS	#5, LIB\$SIGNAL		
				0198	31	00187	BRW	46\$			2641
05	0A	A6			06	E1	0018A	BBC	#6, 10(TFRPSC), 24\$		2644
15	0A	A6			03	E0	0018F	BBS	#3, 10(TFRPSC), 25\$		2645
7E		6B			14	C1	00194	ADDL3	#20, LNK\$GL_CURFIL, -(SP)		2648
					59	DD	00198	PUSHL	R9		
					02	DD	0019A	PUSHL	#2		
			00000000G		8F	DD	0019C	PUSHL	#LINS PSCNXR		
	00000000G	00			04	FB	001A2	CALLS	#4, LIB\$SIGNAL		
		50	04		A4	D0	001A9	MOVL	4(PSCTMAP), MODPSCONTRIBLK		2650
		06			55	E9	001AD	BLBC	WORDPSECTEOM, 26\$		2651
		51	04		A2	D0	001B0	MOVL	4(R2), R1		
					04	11	001B4	BRB	27\$		
		51	03		A2	D0	001B6	MOVL	3(R2), R1		
58		51	08		A0	C1	001BA	ADDL3	8(MODPSCONTRIBLK), R1, TFRADR		2653
		12			55	E9	001BF	BLBC	WORDPSECTEOM, 29\$		2654
					50	D4	001C2	CLRL	R0		
		09	A4		A9	B1	001C4	CMPW	RECLNG, #9		
					02	12	001C8	BNEQ	28\$		
					50	D6	001CA	INCL	R0		
57	08	A2			00	EF	001CC	EXTZV	#0, #1, 8(R2), TFRWEAK		2656
					10	11	001D2	BRB	31\$		
					50	D4	001D4	CLRL	R0		
		08	A4		A9	B1	001D6	CMPW	RECLNG, #8		
					02	12	001DA	BNEQ	30\$		
					50	D6	001DC	INCL	R0		
57	07	A2			00	EF	001DE	EXTZV	#0, #1, 7(R2), TFRWEAK		
		57			57	D2	001E4	MCOML	TFRWEAK, TFRWEAK		
		50			57	CB	001E7	BICL3	TFRWEAK, R0, TFRWEAK		
		50			6B	D0	001EB	MOVL	LNK\$GL_CURFIL, R0		2658
40	0A	A0			05	E0	001EE	BBS	#5, 10(R0), 36\$		
					6A	D5	001F3	TSTL	LNK\$GL_TFRPSC		2660
		33	94		07	13	001F5	BEQL	32\$		
		30			A9	E9	001F7	BLBC	WEAKTFRADR, 35\$		2661
		00			57	E8	001FB	BLBS	TFRWEAK, 35\$		
1B	00000000G	00			06	E0	001FE	BBS	#6, LNK\$GL_CTLMSK+3, 34\$		2666
			00000000G		00	95	00206	TSTB	LNK\$GT_IMGID		2667
					04	13	0020C	BEQL	33\$		
					6A	D5	0020E	TSTL	LNK\$GL_TFRPSC		2668
		50	B8		0F	12	00210	BNEQ	34\$		
					A9	9A	00212	MOVZBL	IMAGEIDSTRING, R0		2670
					50	D6	00216	INCL	R0		
00000000G	00	B8	A9		50	28	00218	MOV3	R0, IMAGEIDSTRING, LNK\$GT_IMGID		2671
		6A			56	D0	00221	MOVL	TFRPSC, LNK\$GL_TFRPSC		2673
		FC	AA		58	D0	00224	MOVL	TFRADR, LNK\$GL_TFRADR		2674
		94	A9		57	90	00228	MOVB	TFRWEAK, WEAKTFRADR		2675

OF	OA	A0		2B	11	0022C		BRB	38\$		2658
				05	E1	0022E	35\$:	BBC	#5, 10(R0), 37\$		2679
			F8	AA	D5	00233	36\$:	TSTL	LNK\$GL_DBGTFPS		2681
				0A	12	00236		BNEQ	37\$		
	F8	AA		56	D0	00238		MOVL	TFRPSC, LNK\$GL_DBGTFPS		2684
	F4	AA		58	D0	0023C		MOVL	TFRADR, LNK\$GL_DBGTFR		2685
				17	11	00240		BRB	38\$		2679
		14		57	E8	00242	37\$:	BLBS	TFRWEAK, 38\$		2689
			14	A0	9F	00245		PUSHAB	20(R0)		2696
				59	DD	00248		PUSHL	R9		
				02	DD	0024A		PUSHL	#2		
		00000000G		8F	DD	0024C		PUSHL	#LIN\$ MULTFR		
00000000G		00		04	FB	00252		CALLS	#4, LIB\$SIGNAL		
			00000000G	00	95	00259	38\$:	TSTB	LNK\$GT_IMGID		2702
				0F	12	0025F		BNEQ	39\$		
		50		A9	9A	00261		MOVZBL	IMAGEIDSTRING, R0		2704
				50	D6	00265		INCL	R0		
00000000G	00	B8		50	28	00267		MOV3	R0, IMAGEIDSTRING, LNK\$GT_IMGID		2706
	08	EC		06	E0	00270	39\$:	BBS	#6, OBMODESC+20, 40\$		2709
				A9	3C	00275		MOVZWL	OBMODESC+18, R6		2710
				56	D6	00279		INCL	R6		
				10	11	0027B		BRB	41\$		
		56		A9	3C	0027D	40\$:	MOVZWL	OBMODESC+18, R6		2711
		56	0100	C6	9E	00281		MOVAB	256(R6), R6		
		56	00000100	8F	C6	00286		DIVL2	#256, R6		
		56		08	C4	0028D	41\$:	MULL2	#8, OBJDESCSIZE		2708
		56		A6	9E	00290		MOVAB	72(R6), OBJDESCSIZE		
				04	AE	00294		PUSHAB	OBJDESC		2712
				56	DD	00297		FUSHL	OBJDESCSIZE		
00000000G		00		02	FB	00299		CALLS	#2, LNK\$ALLOBLK		
		57		AE	D0	002A0		MOVL	OBJDESC, R7		2713
		90		57	D0	002A4		MOVL	R7, @LASTOBJMOD		
		90		57	D0	002A8		MOVL	R7, LASTOBJMOD		2714
67		D8		56	28	002AC		MOV3	OBJDESCSIZE, OBMODESC, (R7)		2715
				57	DD	002B1		PUSHL	R7		2717
			0C	AE	9F	002B3		PUSHAB	OMDNODE		2716
			F135	CF	9F	002B6		PUSHAB	ALLOC OMDNODE		
			F151	CF	9F	002BA		PUSHAB	LNK\$COMPARE_OMD		
			10	AE	D4	002BE		CLRL	16(SP)		2717
			10	AE	9F	002C1		PUSHAB	16(SP)		
			1C	A7	DD	002C4		PUSHL	28(R7)		2716
			08	AA	9F	002C7		PUSHAB	LNK\$GL_OMDTREE		
00000000G		00		07	FB	002CA		CALLS	#7, LIB\$INSERT_TREE		
				67	D4	002D1		CLRL	(R7)		2718
		50		A9	9E	002D3		MOVAB	OBMODESC+32, R0		2720
		50		A9	D1	002D7		CMPL	OBMODESC+36, R0		
				05	12	002DB		BNEQ	42\$		
		24		A7	9E	002DD		MOVAB	32(R7), 36(R7)		2722
		13		A9	E9	002E2	42\$:	BLBC	OMD HAS DBG, 43\$		2729
		50		AA	3C	002E6		MOVZWL	LNK\$GW_PSCDST, R0		2732
		51		A9	3C	002EA		MOVZWL	OBMODESC+18, R1		
		50		51	C0	002EE		ADDL2	R1, R0		
DO	AA			01	A1	002F1		ADDW3	#1, R0, LNK\$GW_PSCDST		
				A9	94	002F6		CLRB	OMD HAS DBG		2733
		50		A9	D0	002F9	43\$:	MOVL	OBMODESC, CURCONTRIBLK		2739
				0D	13	002FD	44\$:	BEQL	45\$		2741
		51		A0	D0	002FF		MOVL	4(CURCONTRIBLK), NEXTCONTRIBLK		2743

LNK_OBJPASS1
V04=000

N 1
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 81
(20)

56	00	04	A0	57	D0	00303	MOVL	R7, 4(CURCONTRIBLK)	: 2744
			50	51	D0	00307	MOVL	NEXTCONTRIBLK, CURCONTRIBLK	: 2745
				F1	11	0030A	BRB	44\$	: 2741
			6E	00	2C	0030C 45\$:	MOVCS	#0, (SP), #0, OBJDESCSIZE, OBMODESC	: 2748
				D8	A9	00311			: 2749
		FC	A9	A9	9E	00313	MOVAB	OBMODESC+32, OBMODESC+36	: 2750
		A0	A9	8F	B0	00318	MOVW	#2048, MAXRECLNG	: 2751
			50	01	D0	0031E	MOVL	#1, R0	: 2752
					04	00321	RET		
				50	D4	00322 46\$:	CLRL	R0	
					04	00324	RET		

; Routine Size: 805 bytes, Routine Base: \$CODE\$ + 126E


```
2276 2753 1 routine countdbg =
2277 2754 2   begin
2278 2755 2   |
2279 2756 2   ROUTINE TO COUNT THE NUMBER OF DEBUG (TYPES 4 AND 5) RECORDS SEEN AND TO
2280 2757 2   TOTAL THE NUMBER OF BYTES CONTAINED FOR USE AS AN ESTIMATE OF THE
2281 2758 2   SIZE OF DEBUG SYMBOL TABLE TO BE WRITTEN AT END OF BINARY
2282 2759 2   IF THE DEBUGGER IS LINKED IN. ALSO IF AN EXECUTABLE IMAGE
2283 2760 2   WITH THE DEBUGGER IS BEING PRODUCED, TURN OFF THE NO BINARY FLAG FOR
2284 2761 2   THIS MODULE (PROVIDED IT IS NOT THE DEBUGGER ITSELF).
2285 2762 2   |
2286 2763 2
2287 2764 2   if not seqchk ()           ! CHECK LEGAL SEQUENCE
2288 2765 2   then
2289 2766 2       return false;
2290 2767 2
2291 2768 2   if .obmodesc [omd$debuger] ! IF THIS IS THE DEBUGGER ITSELF
2292 2769 2   then
2293 2770 2       return true;         ! IGNORE ANY DEBUG RECORDS
2294 2771 2
2295 2772 2   if not .omd_has_dbg and .lnk$gl_ctlmsk [lnk$debug] ! IF DBG REC FLAG IS CLEAR AND THE
2296 2773 2   then                                         ! DEBUGGER IS BEING INCLUDED
2297 2774 2       begin                                     ! IN THE IMAGE,
2298 2775 2       omd_has_dbg = true;                       ! THEN SET FLAG NOW AND
2299 2776 2       lnk$gl_omddst = .lnk$gl_omddst + 1;       ! ADD TO COUNT OF DBG OBJ MODS
2300 2777 2       end;
2301 2778 2
2302 2779 2   lnk$gl_dbggestim = .lnk$gl_dbggestim + .reclng; ! ADD IN THE LENGTH OF THIS RECORD
2303 2780 2   lnk$gw_dbgrecs = .lnk$gw_dbgrecs + 1;         ! AND COUNT THE RECORD
2304 2781 2
2305 2782 2   if .lnk$gl_ctlmsk [lnk$debug] ! IF IT CONTAINS THE DEBUGGER IT NOW HAS
2306 2783 2   or .lnk$gl_ctlmsk [lnk$trace] !
2307 2784 2   then
2308 2785 2       obmodesc [omd$nobin] = false; ! BINARY WE CARE ABOUT
2309 2786 2
2310 2787 2   return true; ! AND THAT IS ALL
2311 2788 1   end;
```

```
001C 00000 COUNTDBG:
ED37 54 00000000' EF 9E 00002 .WORD Save R2,R3,R4 ; 2753
25 53 00000000G 00 9E 00009 MOVAB LNK$GL_OMDDST, R4 ;
06 52 00000000' EF 9E 00010 MOVAB LNK$GL_CTLMSK, R3 ;
A0 CF 00  FB 00017 MOVAB OBMODESC+20, R2 ;
2D 50  E9 0001C CALLS #0, SEQCHK ; 2764
62 05  E0 0001F BLBC R0, 4$ ;
0A A0 A2 E8 00023 BBS #5, OBMODESC+20, 3$ ; 2768
63 06 E1 00027 BLBS OMD_HAS_DBG, 1$ ; 2772
A0 A2 01 90 0002B BBC #6, LNK$GL_CTLMSK, 1$ ;
50 B8 A2 3C 00031 1$: MOVAB #1, OMD_HAS_DBG ; 2775
38 A4 50 C0 00035 INCL LNK$GL_OMDDST ; 2776
05 63 40 A4 B6 00039 MOVZWL RECLNG, R0 ; 2779
06 E0 0003C ADDL2 R0, LNK$GL_DBGESTIM ;
INCW LNK$GW_DBGRECS ; 2780
BBS #6, LNK$GL_CTLMSK, 2$ ; 2782
```


LNK_OBJPASS1
V04=000

C 2
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 83
(21)

03 02 A3
 62
 50

02	E1	00040		BBC	#2, LNK\$GL_CTLMSK+2, 3\$
02	8A	00045	2\$:	BICB2	#2, OBMODESC+20
01	D0	00048	3\$:	MOVL	#1, R0
	04	0004B		RET	
50	D4	0004C	4\$:	CLRL	R0
	04	0004E		RET	

: 2783
: 2785
: 2787
: 2788
:

; Routine Size: 79 bytes, Routine Base: \$CODE\$ + 1593


```
: 2313      2789 1 routine compare_idc (entityname, node, idcrec) =
: 2314      2790 2   begin
: 2315      2791 2   :
: 2316      2792 2   : COMPARE AN ENTITY NAME WITH THE CURRENT NODE
: 2317      2793 2   :
: 2318      2794 2   :
: 2319      2795 2   map
: 2320      2796 2       entityname : ref vector [, byte],
: 2321      2797 2       node : ref block [, byte],
: 2322      2798 2       idcrec : ref block [, byte];
: 2323      2799 2
: 2324      2800 2   return ch$compare (.entityname [0], entityname [1], .node [idcd$b_namlng], node [idcd$t_name], 0)
: 2325      2801 1   end;
```

				001C 00000 COMPARE_IDC:				
			52	04	AC D0 00002	.WORD	Save R2,R3,R4	: 2789
			53		62 9A 00006	MOVL	ENTITYNAME, R2	: 2800
			50	08	AC D0 00009	MOVZBL	(R2), R3	
			51	1E	A0 9A 0000D	MOVL	NODE, R0	
			54		01 D0 00011	MOVZBL	30(R0), R1	
51	00	01	A2		53 2D 00014	MOVL	#1, R4	
				1F	A0 0001A	CMPC5	R3, 1(R2), #0, R1, 31(R0)	
			54		03 1A 0001C	BGTRU	1\$	
			50		01 D9 0001E	SBWC	#1, R4	
					54 D0 00021	MOVL	R4, R0	
					04 00024	RET		: 2801

; Routine Size: 37 bytes, Routine Base: \$CODE\$ + 15E2


```
2327 2802 1 routine alloc_idc (entityname, retnodeadr, idcrec) =
2328 2803 2 begin
2329 2804 2
2330 2805 2 ALLOCATE AND FILL IN A NEW IDCD BLOCK
2331 2806 2
2332 2807 2 map
2333 2808 2     entityname : ref vector [, byte],      ! POINTS TO ASCIC NAME
2334 2809 2     retnodeadr : ref vector [, long],
2335 2810 2     idcrec : ref block [, byte];           ! THE IDC GSD SUB-RECORD
2336 2811 2 bind
2337 2812 2     identstring = idcrec [idc$b_namlng] + 1 + .idcrec [idc$b_namlng] : vector [, byte],
2338 2813 2     ! IDENT STRING IN RECORD
2339 2814 2     binaryident = identstring [1],          ! ADDR OF BINARY IDENT
2340 2815 2     objectname = identstring [1] + .identstring [0] : vector [, byte];      ! ADDR OF OBJECT NAME STRING
2341 2816 2
2342 2817 2 local
2343 2818 2     ptr,
2344 2819 2     idsize,
2345 2820 2     newnode : ref block [, byte];
2346 2821 2
2347 2822 2
2348 2823 2 COMPUTE SIZE OF IDENT STRING
2349 2824 2
2350 2825 2     idsize = 0;
2351 2826 2
2352 2827 2     if not .idcrec [idc$v_binident] then idsize = .identstring [0] + 1;
2353 2828 2
2354 2829 2
2355 2830 2 ALLOCATE A NEW IDCD BLOCK
2356 2831 2
2357 2832 2     lnk$alloblk (idcd$c_size + .entityname [0] + .idsize + .objectname [0] + 1, newnode);
2358 2833 2     !+1 FOR SIZE OF OBJECTNAME IN ASCIC STRING
2359 2834 2     newnode [idcd$w_flags] = .idcrec [idc$w_flags]; ! COPY FLAGS FROM IDC SUBRECORD
2360 2835 2     newnode [idcd$l_defomd] = .lnk$gw_nmodules; ! SET DEFINING MODULE NUMBER
2361 2836 2     newnode [idcd$l_deffdb] = .lnk$gl_curfil; ! DEFINING FILE FDB ADDRESS
2362 2837 2     newnode [idcd$b_idlng] = .identstring [0]; ! LENGTH OF IDENT STRING
2363 2838 2     newnode [idcd$b_objlng] = .objectname [0]; ! LENGTH OF OBJECT NAME STRING
2364 2839 2     newnode [idcd$b_namlng] = .idcrec [idc$b_namlng]; ! ENTITY NAME LENGTH
2365 2840 2     ptr = ch$move (.idcrec [idc$b_namlng], ! COPY IN THE ENTITY NAME
2366 2841 2     idcrec [idc$b_namlng] + 1, newnode [idcd$t_name]);
2367 2842 2     newnode [idcd$l_objnam] = .ptr; ! SET POINTER TO OBJECT NAME
2368 2843 2     ptr = ch$move (.objectname [0] + 1, objectname [0], .ptr); ! MOVE IN THE OBJECT NAME
2369 2844 2
2370 2845 2     if .newnode [idcd$v_binident] ! IF BINARY IDENT, SET IN THE IDENT
2371 2846 2     then
2372 2847 2         newnode [idcd$l_ident] = .binaryident
2373 2848 2     else
2374 2849 2         begin
2375 2850 2             newnode [idcd$l_ident] = .ptr; ! ASCIC IDENT, SET POINTER TO IDENT STRING
2376 2851 2             ch$move (.identstring [0] + 1, identstring [0], .ptr); ! AND COPY IN THE IDENT
2377 2852 2         end;
2378 2853 2
2379 2854 2
2380 2855 2 RETURN NODE ADDRESS TO CALLER
2381 2856 2
2382 2857 2     retnodeadr [0] = .newnode;
2383 2858 2     return true
```


: 2384

```
2859 1      end:
```

F 2
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 86
(23)

03FC 00000 ALLOC_IDC:

		5E		04	C2	00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9	: 2802
		52		OC	AC	D0	SUBL2	#4, SP	:
		50		03	A2	9A	MOVL	IDCREC, R2	: 2812
		57		04	A240	9E	MOVZBL	3(R2), R0	:
		59			67	9A	MOVAB	4(R2)[R0], R7	:
					51	D4	MOVZBL	(R7), R9	: 2815
						E8	CLRL	IDSIZE	: 2825
		04		01	A2	9E	BLBS	1(R2), 1\$	: 2827
		51		01	A9	DD	MOVAB	1(R9), IDSIZE	:
					5E	0001F	PUSHL	SP	: 2832
		50		04	BC	9A	MOVZBL	@ENTITYNAME, R0	:
		50			51	C0	ADDL2	IDSIZE, R0	:
		58		01	A947	9A	MOVZBL	1(R9)[R7], R8	:
				20	A840	9F	PUSHAB	32(R8)[R0]	:
					02	FB	CALLS	#2, LNK\$ALLOBLK	:
					6E	D0	MOVL	NEWNODE, R6	: 2834
		0A	A6	01	A2	B0	MOVW	1(R2), 10(R6)	:
		OC	A6	00000000'	EF	3C	MOVZWL	LNK\$GW_NMODULES, 12(R6)	: 2835
		10	A6	00000000G	00	D0	MOVL	LNK\$GL_CURFIL, 16(R6)	: 2836
		14	A6		59	90	MOVB	R9, 20(R6)	: 2837
		15	A6		58	90	MOVB	R8, 21(R6)	: 2838
		1E	A6		03	A2	MOVB	3(R2), 30(R6)	: 2839
			50	03	A2	9A	MOVZBL	3(R2), R0	: 2840
1F	A6	04	A2		50	28	MOV C3	R0, 4(R2), 31(R6)	: 2841
		1A	A6		53	D0	MOVL	PTR, 26(R6)	: 2842
			50	01	A8	9E	MOVAB	1(R8), R0	: 2843
63		01	A947		50	28	MOV C3	R0, 1(R9)[R7], (PTR)	:
		07		0A	A6	E9	BLBC	10(R6), 2\$	: 2845
		16	A6	01	A7	D0	MOVL	1(R7), 22(R6)	: 2847
					OC	11	BRB	3\$	:
		16	A6		53	D0	MOVL	PTR, 22(R6)	: 2850
			50	01	A9	9E	MOVAB	1(R9), R0	: 2851
63			67		50	28	MOV C3	R0, (R7), (PTR)	:
		08	BC		56	D0	MOVL	R6, @RETNODEADR	: 2857
			50		01	D0	MOVL	#1, R0	: 2858
					04	00093	RET		: 2859

; Routine Size: 148 bytes, Routine Base: \$CODE\$ + 1607


```
2386 1 routine randentity =
2387 2   begin
2388 2   :
2389 2   : THIS ROUTINE PROCESSES THE IDC GSD RECORD.
2390 2   :
2391 2   local
2392 2   status,
2393 2   length,
2394 2   defomd : ref block [, byte],      ! DEFINING OMD ADDRESS
2395 2   deffdb : ref block [, byte],      ! DEFINING FDB ADDRESS
2396 2   identstring : ref vector [, byte], ! POINTER TO IDENT STRING
2397 2   objectname : ref vector [, byte], ! POINTER TO OBJECT NAME STRING
2398 2   entry : ref block [, byte];      ! ADDRESS OF ENTRY
2399 2
2400 2   bind
2401 2   idcrec = objvec [.gsdoffset] : block [, byte]; ! POINT TO THE IDC SUBRECORD
2402 2
2403 2   :
2404 2   : CHECK THE LENGTH OF ALL THE NAME STRINGS
2405 2   :
2406 2   : THE ENTITY NAME
2407 2   :
2408 2   :
2409 2   :
2410 2   if .idcrec [idc$b_namlng] eql 0 or .idcrec [idc$b_namlng] gtru sym$c_maxlng
2411 2   then
2412 2   begin
2413 2   signal (lin$_illnamelen, 6, lnk$gt_entity, idcrec [idc$b_namlng], .idcrec [idc$b_namlng],
2414 2   sym$c_maxlng, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
2415 2   return false
2416 2   end;
2417 2
2418 2   :
2419 2   : THE IDENT STRING
2420 2   :
2421 2   identstring = idcrec [idc$b_namlng] + 1 + .idcrec [idc$b_namlng];
2422 2
2423 2   if .identstring [0] eql 0 or .identstring [0] gtru sym$c_maxlng
2424 2   then
2425 2   begin
2426 2   signal (lin$_illnamelen, 6, lnk$gt_ident, identstring [0], .identstring [0], sym$c_maxlng,
2427 2   obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
2428 2   return false
2429 2   end;
2430 2
2431 2   :
2432 2   : AND THE OBJECT NAME
2433 2   :
2434 2   objectname = identstring [1] + .identstring [0];
2435 2
2436 2   if .objectname [0] eql 0 or .objectname [0] gtru sym$c_maxlng
2437 2   then
2438 2   begin
2439 2   signal (lin$_illnamelen, 6, lnk$gt_objnam, objectname [0], .objectname [0], sym$c_maxlng,
2440 2   obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
2441 2   return false
2442 2   end;
```



```
2443 2917 2
2444 2918
2445 2919
2446 2920
2447 2921
2448 2922
2449 2923
2450 2924
2451 2925
2452 2926
2453 2927
2454 2928
2455 2929
2456 2930
2457 2931
2458 2932
2459 2933
2460 2934
2461 2935
2462 2936
2463 2937
2464 2938
2465 2939
2466 2940
2467 2941
2468 2942
2469 2943
2470 2944
2471 2945
2472 2946
2473 2947
2474 2948
2475 2949
2476 2950
2477 2951
2478 2952
2479 2953
2480 2954
2481 2955
2482 2956
2483 2957
2484 2958
2485 2959
2486 2960
2487 2961
2488 2962
2489 2963
2490 2964
2491 2965
2492 2966
2493 2967
2494 2968
2495 2969
2496 2970
2497 2971
2498 2972
2499 2973

UPDATE GSD OFFSET INTO RECORD

length = objectname [1] + .objectname [0] - idcrec;
gsdoffset = .gsdoffset + .length;

ENTRY APPEARS GOOD. LET'S ENTER IT

if (status = lib$insert_tree (lnk$gl_entitree, idcrec [idc$b_namlng], %ref (0), compare_idc, alloc_idc,
    entry, idcrec))=eql lib$keyalrins
then
begin
lib$lookup_tree (lnk$gl_omdtree, .entry [idcd$l_defomd],          ! FIND DEFINING MODULE OMD
    lnk$compare_omd, defomd);
defomd = .defomd [node$l_ptr];          ! REALLY POINT TO THE OMD
deffdb = .entry [idcd$l_deffdb];        ! GET THE DEFINING FDB

if (.idcrec [idc$v_binident] neq .entry [idcd$v_binident])
    or (.idcrec [idc$v_binident]          ! IF BOTH ARE NOT THE SAME (ASCII OR BINARY)
    and (.idcrec [idc$v_idmatch] neq .entry [idcd$v_idmatch]))
    ! OR BINARY IDENT WITH DIFFERENT MATCH CONTROL
then
signal ((lin$entidmtcht and not sts$m_severity) or .idcrec [idc$v_errsev], 5,
    idcrec [idc$b_namlng],          ! ISSUE WARNING
    obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename], defomd [omd$b_namlng],
    deffdb [fdb$q_filename]);

if not ch$eql (.objectname [0], objectname [1],          ! CHECK THAT OBJECT NAMES THE SAME
    .entry [idcd$b_objlng], .entry [idcd$l_objnam] + 1)
then
signal ((lin$entidmtcho and not sts$m_severity) or .idcrec [idc$v_errsev], 7, objectname [0],
    idcrec [idc$b_namlng], obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename],
    .entry [idcd$l_objnam], defomd [omd$b_namlng], deffdb [fdb$q_filename]);

if not .entry [idcd$v_binident]          ! NOW CHECK THE IDENT MATCH
then
begin
if not ch$eql (.identstring [0], identstring [1],          ! CHECK ASCII IDENTS
    .entry [idcd$b_idlng], .entry [idcd$l_ident] + 1, 0)
then
signal ((lin$entidmtch and not sts$m_severity) or .idcrec [idc$v_errsev], 8, objectname [0],
    identstring [0], idcrec [idc$b_namlng], obmodesc [omd$b_namlng],
    lnk$gl_curfil [fdb$q_filename], .entry [idcd$l_ident], defomd [omd$b_namlng],
    deffdb [fdb$q_filename]);

end
else
begin

CHECK BINARY IDENTS

bind
ident1 = identstring [1] : block [, byte],
```



```
: 2500      2974  4      ident2 = entry [idcd$l_ident] : block [, byte];
: 2501      2975  4
: 2502      2976  4      local
: 2503      2977  4          ident1majid,
: 2504      2978  4          ident1minid,
: 2505      2979  4          ident2majid,
: 2506      2980  4          ident2minid;
: 2507      2981  4
: 2508      2982  4      ident1majid = .ident1 [gmt$b_majorid];      ! BLISS GENERATES BETTER CODE THIS WAY
: 2509      2983  4      ident1minid = .ident1 [gmt$b_minorid];
: 2510      2984  4      ident2majid = .ident2 [gmt$b_majorid];
: 2511      2985  4      ident2minid = .ident2 [gmt$b_minorid];
: 2512      2986  4
: 2513      2987  4      if .ident1majid neg .ident2majid      ! MAJOR IDS MUST BE EQUAL
: 2514      2988  5      or (if .entry [idcd$v_idmatch] eql idc$c_leq      ! COMPARE MINOR IDS BASED ON MATCH CONTROL
: 2515      2989  5      then .ident2minid gtrü .ident1minid else .ident2minid neq .ident1minid)
: 2516      2990  4      then
: 2517      2991  4          signal ((lin$entidmtchb and not sts$m_severity) or .idcrec [idc$v_errsev], 10,
: 2518      2992  4          objectname [0], .ident1majid, .ident1minid, idcrec [idc$b_namlng],
: 2519      2993  4          obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename], .ident2majid, .ident2minid,
: 2520      2994  4          defomd [omd$b_namlng], deffdb [fdb$q_filename]);
: 2521      2995  4
: 2522      2996  3      end;
: 2523      2997  3
: 2524      2998  2      end;
: 2525      2999  2
: 2526      3000  2      return true
: 2527      3001  1      end;
```

```
OFFC 00000 RANDENTITY:
5E      10  C2 00002      .WORD      Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11      : 2860
52 00000000' EF 3C 00005      SUBL2      #16, SP      : 2875
52 00000000' EF C0 0000C      MOVZWL      GSDOFFSET, R2
5B      03  A2 9E 00013      ADDL2      OBJVEC, R2
      6B 95 00017      MOVAB      3(R2), R11      : 2884
      05 13 00019      TSTB      (R11)
1F      6B 91 0001B      BEQL      1$
      1D 1B 0001E      CMPB      (R11), #31
7E 00000000G 00      BLEQU      2$
      14 C1 00020 1$:      ADDL3      #20, LNK$GL_CURFIL, -(SP)      : 2888
      EF 9F 00028      PUSHAB      OBMODESC+40
      1F DD 0002E      PUSHL      #31
7E      6B 9A 00030      MOVZBL      (R11), -(SP)
      5B DD 00033      PUSHL      R11
      00000000' EF 9F 00035      PUSHAB      LNK$GT_ENTITY      : 2887
      5F 11 0003B      BRB      6$      : 2888
50      6B 9A 0003D 2$:      MOVZBL      (R11), R0      : 2895
58      04 A042 9E 00040      MOVAB      4(R0)(R2), IDENTSTRING
      68 95 00045      TSTB      (IDENTSTRING)      : 2897
      05 13 00047      BEQL      3$
1F      68 91 00049      CMPB      (IDENTSTRING), #31
      1D 1B 0004C      BLEQU      4$
7E 00000000G 00      14 C1 0004E 3$:      ADDL3      #20, LNK$GL_CURFIL, -(SP)      : 2901
```


			00000000'	EF	9F	00056	PUSHAB	OBMODESC+40		
				1F	DD	0005C	PUSHL	#31		
	7E			68	9A	0005E	MOVZBL	(IDENTSTRING), -(SP)		
			00000000'	58	DD	00061	PUSHL	IDENTSTRING		
				EF	9F	00063	PUSHAB	LNK\$GT_IDENT		2900
				31	11	00069	BRB	6\$		2901
50	04	AE		68	9A	0006B	MOVZBL	(IDENTSTRING), 4(SP)		2908
59	04	AE		01	C1	0006F	ADDL3	#1, 4(SP), R0		
		58		50	C1	00074	ADDL3	R0, IDENTSTRING, OBJECTNAME		
				69	95	00078	TSTB	(OBJECTNAME)		2910
				05	13	0007A	BEQL	5\$		
		1F		69	91	0007C	CMPB	(OBJECTNAME), #31		
				2D	1B	0007F	BLEQU	7\$		
7E	00000000G	00		14	C1	00081	ADDL3	#20, LNK\$GL_CURFIL, -(SP)		2914
			00000000'	EF	9F	00089	PUSHAB	OBMODESC+40		
				1F	DD	0008F	PUSHL	#31		
		7E		69	9A	00091	MOVZBL	(OBJECTNAME), -(SP)		
				59	DD	00094	PUSHL	OBJECTNAME		
			00000000'	EF	9F	00096	PUSHAB	LNK\$GT_OBJNAM		2913
				06	DD	0009C	PUSHL	#6		2914
			00000000G	8F	DD	0009E	PUSHL	#LINS_ILLNAMELEN		
	00000000G	00		08	FB	000A4	CALLS	#8, LIB\$SIGNAL		
				01AD	31	000AB	BRW	17\$		2915
		53		69	9A	000AE	MOVZBL	(OBJECTNAME), R3		2921
50		59		53	C1	000B1	ADDL3	R3, OBJECTNAME, R0		
		50		52	C2	000B5	SUBL2	R2, R0		
				50	D6	000B8	INCL	LENGTH		
	00000000'	EF		50	A0	000BA	ADDW2	LENGTH, GSDOFFSET		2922
				52	DD	000C1	PUSHL	R2		2927
		0C		AE	9F	000C3	PUSHAB	ENTRY		
		FEA2		CF	9F	000C6	PUSHAB	ALLOC_IDC		
		FE79		CF	9F	000CA	PUSHAB	COMPARE_IDC		
		10		AE	D4	000CE	CLRL	16(SP)		
		10		AE	9F	000D1	PUSHAB	16(SP)		
				5B	DD	000D4	PUSHL	R11		
			00000000'	EF	9F	000D6	PUSHAB	LNK\$GL_ENTITREE		
	00000000G	00		07	FB	000DC	CALLS	#7, LIB\$INSERT_TREE		
	00000000G	8F		50	D1	000E3	CMPB	STATUS, #LIB\$_REYALRINS		2928
				03	13	000EA	BEQL	8\$		
				0168	31	000EC	BRW	16\$		
		0C		AE	9F	000EF	PUSHAB	DEFOMD		2931
		EEEE		CF	9F	000F2	PUSHAB	LNK\$COMPARE_OMD		
		10		AE	D0	000F6	MOVL	ENTRY, R4		
	54			0C	A4	000FA	PUSHL	12(R4)		
			00000000'	EF	9F	000FD	PUSHAB	LNK\$GL_OMDTREE		
	00000000G	00		04	FB	00103	CALLS	#4, LIB\$LOOKUP_TREE		
		50		0C	AE	0010A	MOVL	DEFOMD, R0		2933
		0C		0A	A0	0010E	MOVL	10(R0), DEFOMD		
		55		10	A4	00113	MOVL	16(R4), DEFFDB		2934
		5A		01	A2	00117	MOVAB	1(R2), R10		2936
50	0A	A4		6A	8D	0011B	XORB3	(R10), 10(R4), R0		
		0D		50	E8	00120	BLBS	R0, 9\$		
		38		6A	E9	00123	BLBC	(R10), 10\$		2938
50	0A	A4		6A	8D	00126	XORB3	(R10), 10(R4), R0		2939
		06		50	93	0012B	BITB	R0, #6		
				2E	13	0012E	BEQL	10\$		
			14	A5	9F	00130	PUSHAB	20(DEFFDB)		2944

	7E	10	AE	28	C1	00133	ADDL3	#40, DEFOMD, -(SP)	2943
	7E	00000000G	00	14	C1	00138	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	
			00000000'	EF	9F	00140	PUSHAB	OBMODESC+40-	
				5B	DD	00146	PUSHL	R11	2944
				05	DD	00148	PUSHL	#5	
50	6A		03	03	EF	0014A	EXTZV	#3, #3, (R10), R0	2941
	7E		50	8F	C9	0014F	BISL3	#<LINS ENTIDMTCHT&-8>, R0, -(SP)	
		00000000G	00	07	FB	00157	CALLS	#7, LIB\$SIGNAL	2944
			50	A4	9A	0015E	MOVZBL	21(R4), R0	2947
			56	A4	D0	00162	MOVL	26(R4), R6	
50	00	01	A9	53	2D	00166	CMPC5	R3, 1(OBJECTNAME), #0, R0, 1(R6)	2946
				01	A6	0016C			
				32	13	0016E	BEQL	11\$	
			14	A5	9F	00170	PUSHAB	20(DEFMDB)	2951
	7E	10	AE	28	C1	00173	ADDL3	#40, DEFOMD, -(SP)	
				56	DD	00178	PUSHL	R6	
	7E	00000000G	00	14	C1	0017A	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	2950
			00000000'	EF	9F	00182	PUSHAB	OBMODESC+40-	
			0A00	8F	BB	00188	PUSHR	#*M<R9,R11>	2951
				07	DD	0018C	PUSHL	#7	
50	6A		03	03	EF	0018E	EXTZV	#3, #3, (R10), R0	2949
	7E		50	8F	C9	00193	BISL3	#<LINS ENTIDMTCHO&-8>, R0, -(SP)	
		00000000G	00	09	FB	0019B	CALLS	#9, LIB\$SIGNAL	2951
			57	A4	9E	001A2	MOVAB	22(R4), R7	2958
			48	A4	E8	001A6	BLBS	10(R4), 12\$	
			50	A4	9A	001AA	MOVZBL	20(R4), R0	
			56	67	D0	001AE	MOVL	(R7), R6	
50	00	01	A8	04	AE	001B1	CMPC5	4(SP), 1(IDENTSTRING), #0, R0, 1(R6)	2957
				01	A6	001B8			
				61	13	001BA	BEQL	14\$	
			14	A5	9F	001BC	PUSHAB	20(DEFMDB)	2963
	7E	10	AE	28	C1	001BF	ADDL3	#40, DEFOMD, -(SP)	2962
				56	DD	001C4	PUSHL	R6	2963
	7E	00000000G	00	14	C1	001C6	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	2962
			00000000'	EF	9F	001CE	PUSHAB	OBMODESC+40-	2961
			0900	8F	BB	001D4	PUSHR	#*M<R8,R11>	2963
				59	DD	001D8	PUSHL	OBJECTNAME	
				08	DD	001DA	PUSHL	#8	
50	6A		03	03	EF	001DC	EXTZV	#3, #3, (R10), R0	2960
	7E		50	8F	C9	001E1	BISL3	#<LINS ENTIDMTCH&-8>, R0, -(SP)	
		00000000G	00	0A	FB	001E9	CALLS	#10, LIB\$SIGNAL	2963
				65	11	001F0	BRB	16\$	2953
			50	A8	9E	001F2	MOVAB	1(IDENTSTRING), R0	2973
			53	A0	9A	001F6	MOVZBL	3(R0), IDENT1MAJID	2982
51	60		18	00	EF	001FA	EXTZV	#0, #24, (R0), IDENT1MINID	2983
			52	A7	9A	001FF	MOVZBL	3(R7), IDENT2MAJID	2984
50	67		18	00	EF	00203	EXTZV	#0, #24, (R7), IDENT2MINID	2985
			52	53	D1	00208	CMPL	IDENT1MAJID, IDENT2MAJID	2987
				12	12	0020B	BNEQ	15\$	
			06	A4	93	0020D	BITB	10(R4), #6	2988
				07	12	00211	BNEQ	13\$	
			51	50	D1	00213	CMPL	IDENT2MINID, IDENT1MINID	2989
				3F	1B	00216	BLEQU	16\$	
				05	11	00218	BRB	15\$	
			51	50	D1	0021A	CMPL	IDENT2MINID, IDENT1MINID	
				38	13	0021D	BEQL	16\$	
			14	A5	9F	0021F	PUSHAB	20(DEFMDB)	2994

LNK_OBJPASS1
V04=000

L 2
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 92
(24)

7E	10	AE	28	C1	00222	ADDL3	#40, DEFOMD, -(SP)	:
			50	DD	00227	PUSHL	IDENT2MINID	:
			52	DD	00229	PUSHL	IDENT2MAJID	:
7E	00000000G	00	14	C1	0022B	ADDL3	#20, LNK\$GL_CURFIL, -(SP)	2993
		000000000'	EF	9F	00233	PUSHAB	OBMODESC+40	:
		0802	8F	BB	00239	PUSHR	#*M<R1,R11>	2994
			53	DD	0023D	PUSHL	IDENT1MAJID	:
			59	DD	0023F	PUSHL	OBJECTNAME	:
50			0A	DD	00241	PUSHL	#10	:
6A		03	03	EF	00243	EXTZV	#3, #3, (R10), R0	2991
7E		50	8F	C9	00248	BISL3	#<LIN\$ ENTIDMTCHB&-8>, R0, -(SP)	:
	00000000G	00	0C	FB	00250	CALLS	#12, LTB\$SIGNAL	2994
		50	01	D0	00257	MOVL	#1, R0	3000
				04	0025A	RET		:
			50	D4	0025B	CLRL	R0	3001
				04	0025D	RET		:

; Routine Size: 606 bytes, Routine Base: \$CODE\$ + 169B

```
: 2529      3002 1 routine compare_env (envname, node, envrec) =  
: 2530      3003 2   begin  
: 2531      3004 2   :  
: 2532      3005 2   :   COMPARE AN ENVIRONMENT NAME WITH THE CURRENT NODE IN TREE  
: 2533      3006 2   :  
: 2534      3007 2   :  
: 2535      3008 2   map  
: 2536      3009 2   envname : ref vector [, byte],  
: 2537      3010 2   node : ref block [, byte];  
: 2538      3011 2   bind  
: 2539      3012 2   envdesc = .node + node$c_short : block [, byte];  
: 2540      3013 2   return ch$compare (.envname [0], envname [1], .envdesc [nvd$b_namlng], envdesc [nvd$t_name], 0)  
: 2541      3014 1   end;
```

001C 00000 COMPARE_ENV:									
	50	08	AC	0A	C1	00002	.WORD	Save R2,R3,R4	: 3002
			52	04	AC	D0 00007	ADDL3	#10, NODE, R0	: 3011
			53		62	9A 0000B	MOVL	ENVNAME, R2	: 3013
			51	12	A0	9A 0000E	MOVZBL	(R2), R3	:
			54		01	D0 00012	MOVZBL	18(R0), R1	:
51	00	01	A2		53	2D 00015	MOVL	#1, R4	:
				13	A0	0001B	CMPC5	R3, 1(R2), #0, R1, 19(R0)	:
					03	1A 0001D	BGTRU	1\$	:
			54		01	D9 0001F	SBWC	#1, R4	:
			50		54	D0 00022	MOVL	R4, R0	:
					04	00025	RET		: 3014

; Routine Size: 38 bytes, Routine Base: \$CODE\$ + 18F9


```
: 2543      3015 1 routine alloc_env (envname, retnodeadr, envrec) =
: 2544      3016 2   begin
: 2545      3017 3   :
: 2546      3018 4       ALLOCATE A NEW ENVIRONMENT DESCRIPTOR BLOCK
: 2547      3019 5   :
: 2548      3020 6       map
: 2549      3021 7         envname : ref vector [, byte],
: 2550      3022 8         retnodeadr : ref vector [, long],
: 2551      3023 9         envrec : ref block [, byte];
: 2552      3024 10
: 2553      3025 11       local
: 2554      3026 12         newnode : ref block [, byte],
: 2555      3027 13         envdesc : ref block [, byte];
: 2556      3028 14
: 2557      3029 15       lnk$alloblk (node$c_short + nvd$c_size + .envname [0], newnode);
: 2558      3030 16       envdesc = .newnode + node$c_short;
: 2559      3031 17       ch$fill (0, nvd$c_size, .envdesc);
: 2560      3032 18       envdesc [nvd$l_omdnum] = .lnk$gw_nmodules;
: 2561      3033 19       ch$move (.envname [0] + 1, envname [0], envdesc [nvd$b_namlng]);
: 2562      3034 20       retnodeadr [0] = .newnode;
: 2563      3035 21       return true
: 2564      3036 22   end;
```

				007C 00000 ALLOC_ENV:				
		5E		04	C2 00002	.WORD	Save R2,R3,R4,R5,R6	: 3015
				5E	DD 00005	SUBL2	#4, SP	: 3029
		7E	04	BC	9A 00007	PUSHL	SP	
		6E		1D	C0 0000B	MOVZBL	@ENVNAME, -(SP)	
		00		02	FB 0000E	ADDL2	#29, (SP)	
13	56	00	00000000G	0A	C1 00015	CALLS	#2, LNK\$ALLOBLK	: 3030
		6E		00	2C 00019	ADDL3	#10, NEWNODE, ENVDESC	: 3031
				66	0001E	MOVC5	#0, (SP), #0, #19, (ENVDESC)	
		0C	A6 00000000'	EF	3C 0001F	MOVZWL	LNK\$GW NMODULES, 12(ENVDESC)	: 3032
		50	04	BC	9A 00027	MOVZBL	@ENVNAME, R0	: 3033
				50	D6 0002B	INCL	R0	
	12	A6	04	BC	50 28 0002D	MOVC3	R0, @ENVNAME, 18(ENVDESC)	
		08	BC	6E	D0 00033	MOVL	NEWNODE, @RETNODEADR	: 3034
		50		01	D0 00037	MOVL	#1, R0	: 3035
				04	0003A	RET		: 3036

; Routine Size: 59 bytes, Routine Base: \$CODE\$ + 191F


```
: 2566      3037 1 routine insudfenv (envnode) : novalue =
: 2567      3038 2   begin
: 2568      3039 2   :
: 2569      3040 2   THIS ROUTINE ENTERS AN ENVIRONMENT INTO THE UNDEFINED
: 2570      3041 2   ENVIRONMENT LIST.
: 2571      3042 2   :
: 2572      3043 2   map
: 2573      3044 2   envnode : ref block [, byte];
: 2574      3045 2   builtin
: 2575      3046 2   insque;
: 2576      3047 2
: 2577      3048 2   bind
: 2578      3049 2   envdesc = .envnode + node$c_short : block [, byte];
: 2579      3050 2
: 2580      3051 2   local
: 2581      3052 2   ch_result,
: 2582      3053 2   nxtenv : ref block [, byte];          ! POINTER TO SYMBOL VALUE BLOCK
: 2583      3054 2   :
: 2584      3055 2   FIND THE SPOT TO INSERT THE NEW ENVIRONMENT IN THE UNDEFINED LIST
: 2585      3056 2   :
: 2586      3057 2   :
: 2587      3058 2   nxtenv = lnk$gl_udfenv;          ! SCAN THE UNDEFINED LIST
: 2588      3059 2   :
: 2589      3060 2   while (nxtenv = .nxtenv [nvd$l_udflink]) neq lnk$gl_udfenv do
: 2590      3061 2   begin
: 2591      3062 3   :
: 2592      3063 4   if (ch_result = ch$compare (.nxtenv [nvd$b_namlng],          ! IF GREATER
: 2593      3064 3   nxtenv [nvd$t_name], .envdesc [nvd$b_namlng], envdesc [nvd$t_name])) gtr 0
: 2594      3065 3   then
: 2595      3066 3   exitloop          ! THEN WE ARE ALL DONE
: 2596      3067 3   else
: 2597      3068 3   :
: 2598      3069 3   if .ch_result eql 0          ! IF EQUAL
: 2599      3070 3   then
: 2600      3071 3   return;          ! THEN ALREADY IN LIST, SO RETURN
: 2601      3072 3   :
: 2602      3073 2   end;
: 2603      3074 2   :
: 2604      3075 2   insque (envdesc [nvd$l_udflink],          ! INSERT IN UNDEFINED LIST
: 2605      3076 2   .nxtenv [nvd$l_udb[ink]]);
: 2606      3077 2   lnk$gw_nudfenvs = .lnk$gw_nudfenvs + 1;    ! COUNT THE UNDEFINED ENVIRONMENT
: 2607      3078 2   return;
: 2608      3079 1   end;
```

01FC 00000 INSUDFENV:

		58 00000000'	EF 9E 00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8	: 3037
56	04	AC	0A C1 00009	MOVAB	LNK\$GL_UDFENV, R8	: 3049
		54	68 9E 0000E	ADDL3	#10, ENVNODE, R6	: 3058
		54	64 D0 00011 1\$:	MOVAB	LNK\$GL_UDFENV, NXTENV	: 3060
		50	68 9E 00014	MOVL	(NXTENV), NXTENV	:
		50	54 D1 00017	MOVAB	LNK\$GL_UDFENV, R0	:
			20 13 0001A	CMPL	NXTENV, R0	:
				BEQL	3\$	:

LNK_OBJPASS1
V04=000

C 3
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 96
(27)

50	00	13	51	12	A4	9A	0001C	MOVZBL	18(NXTENV), R1	:	3063
			50	12	A6	9A	00020	MOVZBL	18(R6), R0	:	3064
			55		01	D0	00024	MOVL	#1, R5	:	
			A4		51	2D	00027	CMPC5	R1, 19(NXTENV), #0, R0, 19(R6)	:	
				13	A6		0002D			:	
			55		03	1A	0002F	BGTRU	2\$	:	
			57		01	D9	00031	SBWC	#1, R5	:	
					55	D0	00034	MOVL	R5, CH_RESULT	:	
					03	14	00037	BGTR	3\$	:	
					D6	12	00039	BNEQ	1\$	:	3069
						04	0003B	RET		:	3071
	04	B4			66	0E	0003C	INSQUE	(R6), @4(NXTENV)	:	3076
				40	A8	B6	00040	INCW	LNK\$GW_NUDFENV5	:	3077
					04	00043	RET			:	3079

; Routine Size: 68 bytes, Routine Base: \$CODE\$ + 195A

; 2609 3080 1


```
: 2611 3081 1 global routine lnk$findenvmap (envnum) =
: 2612 3082 2   begin
: 2613 3083 2   :
: 2614 3084 2   THIS ROUTINE RETURNS THE ADDRESS OF THE MAPPING
: 2615 3085 2   TABLE ENTRY FOR ENVIRONMENT NUMBER 'ENVNUM' IN
: 2616 3086 2   THE CURRENT MODULE.
: 2617 3087 2   THE ENVIRONMENT MAPPING TABLE IS AN ARRAY POINTED TO BY
: 2618 3088 2   THE MODULE DESCRIPTOR BLOCK. IT IS INITIALLY
: 2619 3089 2   ALLOCATED WITH SPACE SUFFICIENT FOR 256
: 2620 3090 2   ENTRIES. IF MORE THAN 256 ENVIRONMENTS ARE ENCOUNTERED DURING
: 2621 3091 2   THE PROCESSING OF THE OBJECT MODULE, THE INITIAL TABLE IS
: 2622 3092 2   COPIED OUT TO ANOTHER BLOCK, AND THE MAP TABLE ORIGINAL MAP TABLE
: 2623 3093 2   BECOMES A TABLE OF TABLE ADDRESSES.
: 2624 3094 2   :
: 2625 3095 2   literal
: 2626 3096 2       tablesize = 256*pmt$c_size;           ! SIZE IN BYTES OF A BLOCK
: 2627 3097 2   local
: 2628 3098 2       blockoff,                               ! OFFSET OF ENTRY IN EXTENDED BLOCK
: 2629 3099 2       first_entry : ref block [, byte],
: 2630 3100 2       first_256 : ref block [, byte],       ! address of block of first 256
: 2631 3101 2       mapent : ref block [, byte],         ! ADDRESS IN PSECT MAPPING TABLE
: 2632 3102 2       mapoff : ref block [, byte];        ! ADDRESS OF EXTENDED MAPPING BLOCK
: 2633 3103 2
: 2634 3104 2   if .lnk$gl_curomd [omd$l_envmap] eql 0    ! IF NO TABLE ALLOCATED YET
: 2635 3105 2   then
: 2636 3106 3       begin
: 2637 3107 3         lnk$alloblk (tablesize, lnk$gl_curomd [omd$l_envmap]); ! ALLOCATE IT NOW
: 2638 3108 3         ch$fill (0, tablesize, .lnk$gl_curomd [omd$l_envmap]);
: 2639 3109 2       end;
: 2640 3110 2
: 2641 3111 2   mapent = .lnk$gl_curomd [omd$l_envmap] + .envnum*pmt$c_size;
: 2642 3112 2                                     ! GET ENTRY ADDRESS IN BASE MAPPING TABLE
: 2643 3113 2
: 2644 3114 2   if not .lnk$gl_curomd [omd$v_e256]         ! IF NOT INTO EXTENDED MAP TABLE
: 2645 3115 2     and .envnum lequ 255                   ! AND LEQU 255 ENVIRONMENTS
: 2646 3116 2   then
: 2647 3117 2     return .mapent
: 2648 3118 2   :
: 2649 3119 2   EXTENDED ENVIRONMENT ENTRY
: 2650 3120 2   :
: 2651 3121 2   else
: 2652 3122 3     begin
: 2653 3123 3       blockoff = (.envnum mod 256)*pmt$c_size; ! OFFSET OF ENTRY WITHIN EXTENDED TABLE
: 2654 3124 3       mapent = .lnk$gl_curomd [omd$l_envmap] + ((.envnum)/256)*pmt$c_size;
: 2655 3125 3                                     ! GET ADDRESS OF EXTENDED TABLE ADDRESS
: 2656 3126 3
: 2657 3127 3       if ..mapent neq 0                        ! IF EXTENDED TABLE PRESENT
: 2658 3128 3         and .lnk$gl_curomd [omd$v_e256]    ! AND THERE REALLY IS AN EXTENDED TABLE
: 2659 3129 3       then
: 2660 3130 3         return ..mapent + .blockoff          ! THEN RETURN ENTRY ADDRESS
: 2661 3131 3       else
: 2662 3132 4         begin
: 2663 3133 4           lnk$alloblk (tablesize, mapoff);   ! ALLOCATE EXTENDED TABLE BLOCK
: 2664 3134 4
: 2665 3135 4           if not .lnk$gl_curomd [omd$v_e256] ! IF THIS IS THE FIRST ONE
: 2666 3136 4             then
: 2667 3137 5               begin
```



```
: 2668      3138 5      first_entry = lnk$gl_curomd[omd$l_envmap];
: 2669      3139 5      lnk$alloblk(tablesize, first_256);
: 2670      3140 5      ch$move (tablesize, .lnk$gl_curomd [omd$l_envmap], first_256); ! AND COPY BASE TABLE OUT
: 2671      3141 5      ch$fill (0, tablesize, .lnk$gl_curomd [omd$l_envmap]); ! ZERO BASE TABLE
: 2672      3142 5      first_entry[pmt$l_secpmt] = .first_256;
: 2673      3143 5      lnk$gl_curomd [omd$v_e256] = true; ! FLAG INTO EXTENDED TABLE
: 2674      3144 4      end;
: 2675      3145 4
: 2676      3146 4      ch$fill (0, tablesize, .mapoff); ! ZERO NEW BLOCK
: 2677      3147 4
: 2678      3148 4      !
: 2679      3149 4      ! POINT ENTRY IN BASE TABLE TO NEW BLOCK
: 2680      3150 4
: 2681      3151 4      mapent [pmt$l_secpmt] = .mapoff;
: 2682      3152 4      return .mapoff + .blockoff
: 2683      3153 3      end;
: 2684      3154 3
: 2685      3155 2      end;
: 2686      3156 2
: 2687      3157 1      end;
```

				OFFC 00000	.ENTRY		
				5B 00000000G 00 9E 00002	LNK\$FNDENVMAP, Save R2,R3,R4,R5,R6,R7,R8,-	3081	
				5A 00000000G 00 9E 00009	R9,R10,R11		
				5E 08 C2 00010	LNK\$ALLOBLK, R11		
				50 6A D0 00013	LNK\$GL_CUROMD, R10		
				18 A0 D5 00016	#8, SP		
				17 12 00019	LNK\$GL_CUROMD, R0	3104	
				18 A0 9F 0001B	24(R0)		
				7E 0800 8F 3C 0001E	1\$		
				6B 02 FB 00023	24(R0)	3107	
				50 6A D0 00026	#2048, -(SP)		
0800	8F	00		6E 00 2C 00029	CALLS #2, LNK\$ALLOBLK		
				18 B0 00030	LNK\$GL_CUROMD, R0	3108	
				51 6A D0 00032 1\$:	#0, (SP), #0, #2048, @24(R0)		
				50 04 AC D0 00035	LNK\$GL_CUROMD, R1	3111	
				56 18 B140 7E 00039	ENVNUM, R0		
				0D 15 A1 E8 0003E	@24(R1)[R0], MAPENT		
		000000FF		8F 50 D1 00042	21(R1), 2\$	3114	
				04 1A 00049	R0, #255	3115	
				50 56 D0 0004B	2\$		
				04 0004E	MAPENT, R0	3122	
				50 01 7A 0004F 2\$:	RET		
				8E 00000100 8F 7B 00054	EMUL #1, R0, #0, -(SP)	3123	
				52 03 78 0005D	#256, (SP)+, R2, R2		
				50 00000100 8F C6 00061	#3, R2, BLOCKOFF		
				56 18 B140 7E 00068	#256, R0	3124	
				66 D5 0006D	@24(R1)[R0], MAPENT		
				09 13 0006F	(MAPENT)	3127	
				05 15 A1 E9 00071	3\$		
				66 59 C1 00075	21(R1), 3\$	3128	
				04 00079	BLOCKOFF, (MAPENT), R0	3130	
					RET	3132	

LNK_OBJPASS1
V04=000

F 3
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 99
(28)

				7E	0800	5E	DD	0007A	3\$:	PUSHL	SP	:	3133
				6B		8F	3C	0007C		MOVZWL	#2048, -(SP)	:	
				57		02	FB	00081		CALLS	#2, LNK\$ALLOBLK	:	
				2A	15	6A	D0	00084		MOVL	LNK\$GL_CUROMD, R7	:	3135
				57		A7	E8	00087		BLBS	21(R7), 4\$	:	
						18	C0	0008B		ADDL2	#24, FIRST_ENTRY	:	3138
					04	AE	9F	0008E		PUSHAB	FIRST_256	:	3139
				7E	0800	8F	3C	00091		MOVZWL	#2048, -(SP)	:	
				6B		02	FB	00096		CALLS	#2, LNK\$ALLOBLK	:	
				58		6A	D0	00099		MOVL	LNK\$GL_CUROMD, R8	:	3140
0800	8F	04	BE	18	0800	8F	28	0009C		MOV3	#2048, @24(R8), @FIRST_256	:	
			00			00	2C	000A4		MOV5	#0, (SP), #0, #2048, @24(R8)	:	3141
					18	B8		000AB				:	
				67	04	AE	D0	000AD		MOVL	FIRST_256, (FIRST_ENTRY)	:	3142
				A8		01	88	000B1		BISB2	#1, 2T(R8)	:	3143
0800	8F		00	15		00	2C	000B5	4\$:	MOV5	#0, (SP), #0, #2048, @MAPOFF	:	3146
				6E	00	BE		000BC				:	
				66		6E	D0	000BE		MOVL	MAPOFF, (MAPENT)	:	3151
			50	6E		59	C1	000C1		ADDL3	BLOCKOFF, MAPOFF, R0	:	3152
						04	000C5			RET		:	3157

; Routine Size: 198 bytes, Routine Base: \$CODE\$ + 199E

; 2688 3158 1


```
: 2690      3159 1 routine proenv =
: 2691      3160 1
: 2692      3161 1     THIS ROUTINE PROCESSES ENVIRONMENT DEFINITION/REFERENCE
: 2693      3162 1     SUBRECORDS.
: 2694      3163 1
: 2695      3164 2     begin
: 2696      3165 2     bind
: 2697      3166 2         envrec = objvec [.gsdoffset] : block [, byte];
: 2698      3167 2     builtin
: 2699      3168 2         remque;
: 2700      3169 2     local
: 2701      3170 2         ptr,
: 2702      3171 2         envmap : ref block [, byte],
: 2703      3172 2         envnode : ref block [, byte],
: 2704      3173 2         envdesc : ref block [, byte];
: 2705      3174 2
: 2706      3175 2     if .envrec [env$b_namlng] eql 0 or .envrec [env$b_namlng] gtru sym$c_maxlng
: 2707      3176 2     then
: 2708      3177 3         (signal (lin$ illnamelen, 6, lnk$gt_envstring, envrec [env$b_namlng], .envrec [env$b_namlng],
: 2709      3178 2         sym$c_maxlng, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$g_filename]); return false);
: 2710      3179 2
: 2711      3180 2
: 2712      3181 2     INSERT ENVIRONMENT RECORD INTO ENVIRONMENT TREE
: 2713      3182 2
: 2714      3183 2         lib$insert_tree (lnk$gl_envtree, envrec [env$b_namlng], %ref (0), compare_env, alloc_env,
: 2715      3184 2         envnode, envrec);
: 2716      3185 2         envdesc = .envnode + node$c_short;
: 2717      3186 2
: 2718      3187 2     PUT ENVIRONMENT INTO THE ENVIRONMENT MAP TABLE
: 2719      3188 2
: 2720      3189 2         obmodesc [omd$v_noenv] = false;
: 2721      3190 2         obmodesc [omd$w_hienv] = .obmodesc [omd$w_hienv] + 1;
: 2722      3191 2         envmap = lnk$findenvmap (.obmodesc [omd$w_hienv]);
: 2723      3192 2         envmap [pmt$l_pscdes] = .envnode;
: 2724      3193 2
: 2725      3194 2     ! IF .ENVMAP[PMT$L_SYMLST] NEQ 0                                ! SYMBOLS DEFINED BEFORE THE ENVIRONMENT?
: 2726      3195 2     THEN BEGIN                                           ! **ONLY NEED TO ASSOCIATE SYMBOLS WITH ENVIRONMENT FOR CROSS REF
: 2727      3196 2         TRUE
: 2728      3197 2         END;
: 2729      3198 2
: 2730      3199 2     IF THIS IS A DEFINITION, REMOVE FROM UNDEFINED LIST IF IT'S
: 2731      3200 2     ON IT.  IF IT'S THE FIRST REFERENCE TO THIS ENVIRONMENT,
: 2732      3201 2     THEN PUT IT ON THE UNDEFINED LIST.
: 2733      3202 2
: 2734      3203 2
: 2735      3204 2     if .envrec [env$v_def]
: 2736      3205 2     then
: 2737      3206 3         begin
: 2738      3207 3
: 2739      3208 3         if .envdesc [nvd$l_udflink] neq 0
: 2740      3209 3         then
: 2741      3210 4             (remque (envdesc [nvd$l_udflink], ptr);
: 2742      3211 3             lnk$gw_nudfenvs = .lnk$gw_nudfenvs - 1);
: 2743      3212 3
: 2744      3213 3         envdesc [nvd$v_def] = true;
: 2745      3214 3
: 2746      3215 3         if .envmap[pmt$l_symlst] neq 0
```



```

: 2747      3216 3      then
: 2748      3217 4      begin
: 2749      3218 4      envdesc[nvd$l_symlbl] = .envmap[pmt$l_symlst];
: 2750      3219 4      envmap[pmt$l_symlst] = 0;
: 2751      3220 3      end;
: 2752      3221 3      end
: 2753      3222 2      else
: 2754      3223 2
: 2755      3224 2      if not .envdesc [nvd$v_def] and .envdesc [nvd$l_udflink] eql 0
: 2756      3225 2      then
: 2757      3226 2      insudfenv (.envnode);
: 2758      3227 2
: 2759      3228 2      gsdoffset = .gsdoffset + envrec [env$t_name] - envrec [env$b_gsdtyp] + .envrec [env$b_namlng];
: 2760      3229 2      return true
: 2761      3230 1      end;

```

```

                                003C 00000 PROENV: .WORD      Save R2,R3,R4,R5      : 3159
55      FEB5      CF 9E 00002      MOVAB      ALLOC ENV, R5
54 00000000'      EF 9E 00007      MOVAB      GSDOFFSET, R4
5E      08 C2 0000E      SUBL2      #8, SP
53      64 3C 00011      MOVZWL      GSDOFFSET, R3      : 3166
53      0C A4 C0 00014      ADDL2      OBJVEC, R3
50      05 A3 9A 00018      MOVZBL      5(R3), R0      : 3175
1F      05 13 0001C      BEQL      1$
1F      50 91 0001E      CMPB      R0, #31
7E 00000000G 00      2A 1B 00021      BLEQU      2$
                                14 C1 00023 1$:      ADDL3      #20, LNK$GL_CURFIL, -(SP)      : 3178
                                64 A4 9F 0002B      PUSHAB      OBMODESC+40
                                1F DD 0002E      PUSHL      #31
                                50 DD 00030      PUSHL      R0
                                05 A3 9F 00032      PUSHAB      5(R3)      : 3177
                                00000000'      EF 9F 00035      PUSHAB      LNK$GT_ENVSTRING
                                06 DD 0003B      PUSHL      #6      : 3178
                                00000000G 00      8F DD 0003D      PUSHL      #LIN$ ILLNAMELEN
                                08 FB 00043      CALLS      #8, LIB$SIGNAL
                                0084 31 0004A      BRW      6$
                                53 DD 0004D 2$:      PUSHL      R3      : 3183
                                08 AE 9F 0004F      PUSHAB      ENVNODE
                                55 DD 00052      PUSHL      R5
                                DA A5 9F 00054      PUSHAB      COMPARE_ENV
                                10 AE D4 00057      CLRL      16(SP)
                                10 AE 9F 0005A      PUSHAB      16(SP)
                                05 A3 9F 0005D      PUSHAB      5(R3)
                                00000000'      EF 9F 00060      PUSHAB      LNK$GL ENVTREE
52 00000000G 00      07 FB 00066      CALLS      #7, LIB$INSERT TREE      : 3185
                                04 AE      0A C1 0006D      ADDL3      #10, ENVNODE, ENVDESC
                                50 A4      8F 8A 00072      BICB2      #128, OBMODESC+20      : 3189
                                52 A4 B6 00077      INCW      OBMODESC+22      : 3190
                                7E 52 A4 3C 0007A      MOVZWL      OBMODESC+22, -(SP)      : 3191
                                7F A5 01 FB 0007E      CALLS      #1, LNK$FNDENVMAP
                                60 04 AE D0 00082      MOVL      ENVNODE, (ENVMAP)      : 3192
                                20 01 A3 E9 00086      BLBC      1(R3), 4$      : 3204
                                62 D5 0008A      TSTL      (ENVDESC)      : 3208

```


LNK_OBJPASS1
V04=000

1 3
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 102
(29)

			09	13	0008C	BEQL	3\$		
	51		62	0F	0008E	REMQUE	(ENVDESC), PTR	:	3210
		00000000	EF	B7	00091	DECW	LNK\$GW,NUDFENV	:	3211
10	A2		01	88	00097	BISB2	#1, 16(ENVDESC)	:	3213
		04	A0	D5	0009B	TSTL	4(ENVMAP)	:	3215
			19	13	0009E	BEQL	5\$	:	
08	A2	04	A0	D0	000A0	MOVL	4(ENVMAP), 8(ENVDESC)	:	3218
		04	A0	D4	000A5	CLRL	4(ENVMAP)	:	3219
			0F	11	000A8	BRB	5\$	:	3204
	0B	10	A2	E8	000AA	BLBS	16(ENVDESC), 5\$	:	3224
			62	D5	000AE	TSTL	(ENVDESC)	:	
			07	12	000B0	BNEQ	5\$	:	
		04	AE	DD	000B2	PUSHL	ENVNODE	:	3226
3B	A5		01	FB	000B5	CALLS	#1, INSUDFENV	:	
	50		64	3C	000B9	MOVZWL	GSDOFFSET, R0	:	3228
	50		53	C0	000BC	ADDL2	R3, R0	:	
	50		53	C2	000BF	SUBL2	R3, R0	:	
	51	05	A3	9A	000C2	MOVZBL	5(R3), R1	:	
	50		51	C0	000C6	ADDL2	R1, R0	:	
64	50		06	A1	000C9	ADDW3	#6, R0, GSDOFFSET	:	
	50		01	D0	000CD	MOVL	#1, R0	:	3229
				04	000D0	RET		:	
			50	D4	000D1	CLRL	R0	:	3230
				04	000D3	RET		:	

; Routine Size: 212 bytes, Routine Base: \$CODE\$ + 1A64

```
: 2763      3231 1 routine localsymbols =
: 2764      3232 2 begin
: 2765      3233 2 local
: 2766      3234 2     length;
: 2767      3235 2 bind
: 2768      3236 2     symbolrec = objvec [.gsdoffset] : block [, byte];
: 2769      3237 2
: 2770      3238 2 entrymask = 0;                                ! ZERO THE ENTRY MASK SINCE NONE EXISTS
: 2771      3239 2
: 2772      3240 2 if not .symbolrec [lsy$v_def]                ! IF NOT DEFINED
: 2773      3241 2 then begin
: 2774      3242 2     symbolstring = symbolrec [lsrf$b_namlng]; ! POINT TO THE SYMBOL STRING
: 2775      3243 2     length      = symbolrec [lsrf$t_name] - symbolrec [lsrf$t_start] + .symbolrec [lsrf$b_namlng];
: 2776      3244 2     end
: 2777      3245 2 else begin
: 2778      3246 2     symbolstring = symbolrec [lsdf$b_namlng]; ! POINT TO THE SYMBOL
: 2779      3247 2     length      = symbolrec [lsdf$t_name] - symbolrec [lsdf$t_start] + .symbolrec [lsdf$b_namlng];
: 2780      3248 2     end;
: 2781      3249 2
: 2782      3250 2 if not prosymbol (0)                        ! GO PROCESS THIS SYMBOL
: 2783      3251 2 then return false;                          ! AND EXIT IF FAILURE
: 2784      3252 2
: 2785      3253 2 gsdoffset = .gsdoffset + .length;          ! UPDATE THE GSD OFFSET FOR NEXT
: 2786      3254 2 return true
: 2787      3255 1 end;
```

				000C 00000 LOCALSYMBOLS:			
		53	00000000'	EF 9E 00002	.WORD	Save R2,R3	: 3231
		52		63 3C 00009	MOVAB	GSDOFFSET, R3	
		52	0C	A3 C0 0000C	MOVZWL	GSDOFFSET, R2	: 3236
			02	A3 B4 00010	ADDL2	OBJVEC, R2	
16	02	A2		01 E0 00013	CLRW	ENTRYMASK	: 3238
	10	A3	06	A2 9E 00018	BBS	#1, 2(R2), 1\$	: 3240
50		52		52 C3 0001D	MOVAB	6(R2), SYMBOLSTRING	: 3242
		51	06	A2 9A 00021	SUBL3	R2, R2, R0	: 3243
		50		51 C0 00025	MOVZBL	6(R2), R1	
		52	07	A0 9E 00028	ADDL2	R1, R0	
			14	11 0002C	MOVAB	7(R0), LENGTH	
					BRB	2\$	: 3240
	10	A3	0C	A2 9E 0002E 1\$:	MOVAB	12(R2), SYMBOLSTRING	: 3246
50		52		52 C3 00033	SUBL3	R2, R2, R0	: 3247
		51	0C	A2 9A 00037	MOVZBL	12(R2), R1	
		50		51 C0 0003B	ADDL2	R1, R0	
		52	0D	A0 9E 0003E	MOVAB	13(R0), LENGTH	
			7E	D4 00042 2\$:	CLRL	-(SP)	: 3250
F116	CF			01 FB 00044	CALLS	#1, PROSYMBOL	
	07			50 E9 00049	BLBC	R0, 3\$	
	63			52 A0 0004C	ADDW2	LENGTH, GSDOFFSET	: 3253
	50			01 D0 0004F	MOVL	#1, R0	: 3254
				04 00052	RET		
			50	D4 00053 3\$:	CLRL	R0	: 3255
				04 00055	RET		

LNK_OBJPASS1
V04=000

K 3
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 104
(30)

; Routine Size: 86 bytes, Routine Base: \$CODE\$ + 1B38

```
: 2789      3256 1 routine localentpnt =
: 2790      3257 2 begin
: 2791      3258 2 local
: 2792      3259 2     length;
: 2793      3260 2 bind
: 2794      3261 2     symbolrec = objvec [.gsdoffset] : block [, byte];
: 2795      3262 2
: 2796      3263 2     entrymask = .symbolrec [lepm$w_mask];           ! EXTRACT THE ENTRY POINT MASK
: 2797      3264 2     symbolstring = symbolrec [lepm$b_namlng];      ! POINT TO THE SYMBOL
: 2798      3265 2     length = symbolrec [lepm$t_name] - symbolrec [lepm$t_start] + .symbolrec [lepm$b_namlng];
: 2799      3266 2
: 2800      3267 2     if not prosymbol (1)                             ! GO PROCESS THE SYMBOL
: 2801      3268 2     then return false;                             ! AND EXIT IF FAILURE
: 2802      3269 2
: 2803      3270 2     gsdoffset = .gsdoffset + .length;              ! ELSE UPDATE THE OFFSET FOR NEXT
: 2804      3271 2     return true
: 2805      3272 1 end;
```

000C 00000 LOCALENTPNT:							
		53	00000000'	EF 9E 00002	.WORD	Save R2,R3	: 3256
		52		63 3C 00009	MOVAB	GSDOFFSET, R3	
		52		A3 C0 0000C	MOVZWL	GSDOFFSET, R2	: 3261
		52	0C	A3 C0 0000C	ADDL2	OBJVEC, R2	
	02	A3	0C	A2 B0 00010	MOVW	12(R2), ENTRYMASK	: 3263
	10	A3	0E	A2 9E 00015	MOVAB	14(R2), SYMBOLSTRING	: 3264
50		52		52 C3 0001A	SUBL3	R2, R2, R0	: 3265
		51	0E	A2 9A 0001E	MOVZBL	14(R2), R1	
		50		51 C0 00022	ADDL2	R1, R0	
		52	0F	A0 9E 00025	MOVAB	15(R0), LENGTH	
				01 DD 00029	PUSHL	#1	: 3267
	F0D9	CF		01 FB 0002B	CALLS	#1, PROSYMBOL	
		07		50 E9 00030	BLBC	R0, 1\$	
		63		52 A0 00033	ADDW2	LENGTH, GSDOFFSET	: 3270
		50		01 D0 00036	MOVL	#1, R0	: 3271
				04 00039	RET		
			50	D4 0003A 1\$:	CLRL	R0	: 3272
				04 0003C	RET		

; Routine Size: 61 bytes, Routine Base: \$CODE\$ + 1B8E


```
: 2807      3273 1 routine localprocedef =
: 2808      3274 2   begin
: 2809      3275 2   !
: 2810      3276 2       PROCESS PROCEDURE DEFINITION OF MODULE LOCAL SYMBOL
: 2811      3277 2       SEE PROCEDEF FOR AN EXPLANATION OF WHAT A PROCEDURE
: 2812      3278 2       DEFINITION LOOKS LIKE.
: 2813      3279 2   !
: 2814      3280 2   !
: 2815      3281 2   local
: 2816      3282 2       entflag,
: 2817      3283 2       argvaldata : ref vector [, byte],
: 2818      3284 2       argcount;
: 2819      3285 2   !
: 2820      3286 2   !
: 2821      3287 2   !
: 2822      3288 2   if not localentpnt () then return false;      ! PROCESS THE ENTRY POINT SYMBOL PART
: 2823      3289 2   !
: 2824      3290 2   entflag = .syntabent neq 0;                    ! SEE IF ENTERED
: 2825      3291 2   begin
: 2826      3292 2   !
: 2827      3293 2   bind
: 2828      3294 2       formals = objvec [.gsdoffset] : block [, byte]; ! THE FORMAL DESCRIPTION
: 2829      3295 2   !
: 2830      3296 2   if .formals [fml$b_minargs] gtru .formals [fml$b_maxargs] ! IF THERE IS AN INCONSISTENT
: 2831      3297 2   then
: 2832      3298 2       begin
: 2833      3299 2         signal (lin$_illfmlcnt, 5,                ! DESCRIPTION OF FORMALS
: 2834      3300 2         .formals [fml$b_minargs], .formals [fml$b_maxargs], symentnam [snb$b_namlng],
: 2835      3301 2         obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]); ! OUTPUT ERROR MESSAGE
: 2836      3302 2       return false;
: 2837      3303 2       end;
: 2838      3304 2   !
: 2839      3305 2   gsdoffset = .gsdoffset + fml$c_size;          ! UPDATE RECORD POINTER
: 2840      3306 2   !
: 2841      3307 2   if .entflag
: 2842      3308 2       and (argvaldata = .syntabent [sym$l_valdata]) neq 0
: 2843      3309 2   then
: 2844      3310 2       begin
: 2845      3311 2         lnk$dealblk (.argvaldata [0], argvaldata [0]); ! MUST BE A NEW DEFINITION
: 2846      3312 2         symtabent [sym$l_valdata] = 0;                ! SO DISPOSE OF OLD DATA
: 2847      3313 2         end;
: 2848      3314 2       ! AND FORGET IT
: 2849      3315 2   !
: 2850      3316 2   if (argcount = .formals [fml$b_maxargs]) eql 0 ! IF MAXIMUM NUMBER ARGUMENTS
: 2851      3317 2   then
: 2852      3318 2       return true;
: 2853      3319 2       ! IS ZERO, ALL DONE
: 2854      3320 2   !
: 2855      3321 2   if .entflag
: 2856      3322 2   then
: 2857      3323 2       begin
: 2858      3324 2         lnk$alloblk (.argcount + 2, symtabent [sym$l_valdata]); ! ALLOCATE A VALIDATION DATA BLOCK
: 2859      3325 2         argvaldata [0] = .argcount + 2;                ! SET ITS SIZE
: 2860      3326 2         argvaldata [1] = .formals [fml$b_minargs];      ! SAVE MINIMUM ARGUMENT SPECIFICATION
: 2861      3327 2       end;
: 2862      3328 2   !
: 2863      3329 2   incr i from 1 to .argcount
: 2864      3330 2   do
: 2865      3331 2       begin
: 2866      3332 2         ! BEGIN A LOOP THROUGH
: 2867      3333 2         ! ALL FORMAL ARGUMENT DESCRIPTORS
```

```
: 2864      3330  4
: 2865      3331  4
: 2866      3332  4
: 2867      3333  4
: 2868      3334  4
: 2869      3335  4
: 2870      3336  4
: 2871      3337  4
: 2872      3338  4
: 2873      3339  3
: 2874      3340  3
: 2875      3341  3
: 2876      3342  2
: 2877      3343  1

bind
    argdesc = objvec [.gsdoffset] : block [, byte];      ! POINT TO NEXT DESCRIPTOR
if .entflag then argvaldata [.i + 1] = .argdesc [arg$v_passmech];

                                ! EXTRACT THE PASSING MECHANISM
gsdoffset = .gsdoffset + .argdesc [arg$b_bytecnt] +      ! AND UPDATE RECORD POINTER
arg$c_size;
end;

return true;
end;
end;
```

```
007C 00000 LOCALPROCEDEF:
      56 00000000' EF 9E 00002      .WORD      Save R2,R3,R4,R5,R6      : 3273
      B6 AF      00 FB 00009      MOVAB      GSDOFFSET, R6
      3F      50 E9 0000D      CALLS      #0, LOCALENTPNT      : 3288
      14      50 D4 00010      BLBC      R0, 2$
      02      A6 D5 00012      CLRL      R0      : 3290
      55      02 13 00015      TSTL      SYMTABENT
      53      50 D6 00017      BEQL      1$
      53      50 D0 00019 1$:      INCL      R0
      01 A3      66 3C 0001C      MOVL      R0, ENTFLAG
      0C      A6 C0 0001F      MOVZWL     GSDOFFSET, R3      : 3294
      63 91 00023      ADDL2     OBJVEC, R3
      28 1B 00027      CMPB      (R3), 1(R3)      : 3296
      7E 00000000G 00      14 C1 00029      BLEQU     3$
      64      A6 9F 00031      ADDL3     #20, LNK$GL_CURFIL, -(SP)      : 3301
      7E      18 A6      04 C1 00034      PUSHAB   OBMODESC+40
      7E      01 A3 9A 00039      ADDL3     #4, SYMENTNAM, -(SP)      : 3300
      7E      63 9A 0003D      MOVZBL    1(R3), -(SP)      : 3301
      05 DD 00040      MOVZBL    (R3), -(SP)
      00000000G 00 00000000G 05 DD 00040      PUSHL     #5
      8F DD 00042      PUSHL     #LINS_ILLFMLCNT
      07 FB 00048      CALLS     #7, LIB$SIGNAL
      74 11 0004F 2$:      BRB      10$      : 3302
      66      02 A0 00051 3$:      ADDW2     #2, GSDOFFSET      : 3305
      1D      55 E9 00054      BLBC      ENTFLAG, 4$      : 3307
      50      14 A6 D0 00057      MOVL      SYMTABENT, R0      : 3308
      52      18 A0 D0 0005B      MOVL      24(R0), ARGVALDATA
      13 13 0005F      BEQL      4$
      52 DD 00061      PUSHL     ARGVALDATA      : 3311
      7E      62 9A 00063      MOVZBL    (ARGVALDATA), -(SP)
      00000000G 00      02 FB 00066      CALLS     #2, LNK$DEALBLK
      50      14 A6 D0 0006D      MOVL      SYMTABENT, R0      : 3312
      54      18 A0 D4 00071      CLRL      24(R0)
      01 A3 9A 00074 4$:      MOVZBL    1(R3), ARGCOUNT      : 3315
      47 13 00078      BEQL      9$
      7E      14 1A      55 E9 0007A      BLBC      ENTFLAG, 5$      : 3319
      18 C1 0007D      ADDL3     #24, SYMTABENT, -(SP)      : 3322
      02 A4 9F 00082      PUSHAB    2(ARGCOUNT)
```


LNK_OBJPASS1
V04=000

B 4
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 108
(32)

00000000G	00	02	FB	00085	CALLS	#2, LNK\$ALLOBLK	:	
	50	02	A4	9E	0008C	MOVAB	2(ARGCOUNT), R0	: 3323
	62		50	90	00090	MOVB	R0, (ARGVALDATA)	:
01	A2		63	90	00093	MOVB	(R3), 1(ARGVALDATA)	: 3324
			51	D4	00097	CLRL	I	: 3332
			22	11	00099	BRB	8\$	:
	50		66	3C	0009B	MOVZWL	GSDOFFSET, R0	:
	50	0C	A6	C0	0009E	ADDL2	OBJVEC, R0	:
	0A		55	E9	000A2	BLBC	ENTFLAG, 7\$	: 3334
53			00	EF	000A5	EXTZV	#0, #2, (R0), R3	:
	02		53	90	000AA	MOVB	R3, 1(I)[ARGVALDATA]	:
01	A142		66	3C	000AF	MOVZWL	GSDOFFSET, R3	: 3337
	53	01	A0	9A	000B2	MOVZBL	1(R0), R0	:
	50		53	C0	000B6	ADDL2	R3, R0	:
	50		02	A1	000B9	ADDW3	#2, R0, GSDOFFSET	:
66			54	F3	000BD	AOBLEQ	ARGCOUNT, 1, 6\$	: 3327
DA			01	D0	000C1	MOVL	#1, R0	: 3341
	51			04	000C4	RET		:
	50		50	D4	000C5	CLRL	R0	: 3343
			04	000C7	RET			:

; Routine Size: 200 bytes, Routine Base: \$CODE\$ + 1BCB


```
2879 3344 1 routine progsd =
2880 3345 2   begin
2881 3346 2   !
2882 3347 2   !
2883 3348 2   ++
2884 3349 2   VERIFY GSD RECORDS AND DISPATCH ON THE FOUR SUB-TYPES:
2885 3350 2   (0) P-SECTION DEFINITION
2886 3351 2   (1) SYMBOL DEFINITION/REFERENCE
2887 3352 2   (2) ENTRY POINT DEFINITION
2888 3353 2   (3) PROCEDURE DECLARATION
2889 3354 2   --
2890 3355 2   !
2891 3356 2   !
2892 3357 2   !
2893 3358 2   bind
2894 3359 2   gsddispatch = plit (
2895 3360 2   propsectdef,
2896 3361 2   symbols,
2897 3362 2   entpnts,
2898 3363 2   procedef,
2899 3364 2   symbols,
2900 3365 2   entpnts,
2901 3366 2   procedef,
2902 3367 2   randentity,
2903 3368 2   proenv,
2904 3369 2   localsymbols,
2905 3370 2   localentpnt,
2906 3371 2   localprocedef,
2907 3372 2   propsectdef) : vector;
2908 3373 2   !
2909 3374 2   !
2910 3375 2   !
2911 3376 2   local
2912 3377 2   gsdtype;
2913 3378 2   !
2914 3379 2   !
2915 3380 2   !
2916 3381 2   if not seqchk () then return false;
2917 3382 2   !
2918 3383 2   gsdoffset = obj$c_subtyp;
2919 3384 2   !
2920 3385 2   !
2921 3386 2   while .gsdoffset lssu .reclng do
2922 3387 3   begin
2923 3388 3   !
2924 3389 3   if (gsdtype = .objvec [.gsdoffset]) gequ .gsddispatch [-1]
2925 3390 3   then
2926 3391 4   begin
2927 3392 4   signal (lin$gsdtyp, 3, .gsdtype, obmodesc [omd$b_namlng], lnk$gl_curfil [fdb$q_filename]);
2928 3393 4   return false;
2929 3394 4   end
2930 3395 3   else
2931 3396 4   begin
2932 3397 6   wordpsectgsd = ((.gsdtype gequ gsd$c_symw) ! TEST FOR WORD OF PSECT NUMBER
2933 3398 4   and (.gsdtype lequ gsd$c_prow));
2934 3399 6   lclsymgsd = ((.gsdtype gequ gsd$c_lsy) ! TEST FOR LOCAL SYMBOL
2935 3400 4   and (.gsdtype lequ gsd$c_lpro));
```



```

: 2936      3401  4
: 2937      3402  4      if not (.gsddispatch [.gsdtype]) () then return false;
: 2938      3403  4
: 2939      3404  3      end;
: 2940      3405  3
: 2941      3406  2      end;
: 2942      3407  2
: 2943      3408  2      return true
: 2944      3409  1      end;

```

.PSECT \$PLITS\$,NOWRT,NOEXE,2

```

00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 0006F
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00070
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00074 P.AAK: .BLKB 1
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 0008C .LONG 13
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 000A4 .ADDRESS PROPSECTDEF, SYMBOLS, ENTPNTS, PROCEDEF, -
                                                                SYMBOLS, ENTPNTS, PROCEDEF, RANDENTITY, -
                                                                PROENV, LOCALSYMBOLS, LOCALENTPNT, -
                                                                LOCALPROCEDEF, PROPSECTDEF

```

GSDDISPATCH= P.AAK

.PSECT \$CODE\$,NOWRT,2

```

          54 00000000' 001C 00000 PROGSD: .WORD Save R2,R3,R4      : 3344
E645      CF      00 FB 00009 MOVAB   GSDOFFSET, R4      :
          7C      50 E9 0000E CALLS   #0, SEQCHK      : 3381
          64      01 B0 00011 BLBC    R0, 7$      :
          08 A4      64 B1 00014 1$:  MOVW   #1, GSDOFFSET      : 3383
          50      75 1E 00018 BGEQU   8$      : 3386
          50      64 3C 0001A MOVZWL  GSDOFFSET, RECLNG      :
          52      60 9A 00021 ADDL2   GSDOFFSET, R0      : 3389
          EF      52 D1 00024 MOVZBL  (R0), GSdtype      :
          00000000' EF      1E 1F 0002B CMPL   GSdtype, GSDDISPATCH-4      :
          7E 00000000G 00      14 C1 0002D BLSSU   2$      :
          64      A4 9F 00035 ADDL3   #20, LNK$GL_CURFIL, -(SP)      : 3392
          52      03 DD 00038 PUSHAB  OBMODESC+40      :
          00000000G 00      8F DD 0003C PUSHL   GSdtype      :
          05      48 11 00049 PUSHL   #3      :
          51      51 D4 0004B 2$:  PUSHL   #LINS GSdtyp      :
          04      52 D1 0004D BRB     #5, LIB$SIGNAL      :
          02      51 D6 00052 CLRL    R1      : 3393
          50      52 D1 00050 CMPL   GSdtype, #4      : 3397
          06      51 D6 00052 BLSSU   3$      :
          52      50 D4 00054 3$:  INCL    R1      :
          02      52 D1 00056 CLRL    R0      : 3398
          50      02 1A 00059 CMPL   GSdtype, #6      :
          53      51 D2 0005D 4$:  BGTRU   4$      :
          FA A4      50      51 D2 0005D INCL    R0      :
          09      53 8B 00060 MCOML   R1, R3      :
          51      51 D4 00065 BICB3   R3, R0, WORDPSECTGSD      :
          52      52 D1 00067 CLRL    R1      : 3399
          52      52 D1 00067 CMPL   GSdtype, #9

```

LNK_OBJPASS1
V04=000

E 4
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 111
(33)

		02	1F	0006A	BLSSU	5\$		
		51	D6	0006C	INCL	R1		
		50	D4	0006E	CLRL	R0		3400
	0B	52	D1	00070	CMPL	GSDTYPE, #11		
		02	1A	00073	BGTRU	6\$		
		50	D6	00075	INCL	R0		
	53	51	D2	00077	MCOML	R1, R3		
	50	53	8B	0007A	BICB3	R3, R0, LCLSYMGS		
	50	42	D0	0007F	MOVL	GSDDISPATCH[GSDTYPE], R0		3402
	60	00	FB	00087	CALLS	#0, (R0)		
	87	50	E8	0008A	BLBS	R0, 1\$		
		04	11	0008D	BRB	9\$		
	50	01	D0	0008F	MOVL	#1, R0		3408
			04	00092	RET			
		50	D4	00093	CLRL	R0		3409
			04	00095	RET			

; Routine Size: 150 bytes, Routine Base: \$CODE\$ + 1C93

; 2945 3410 1


```
2947 3411 1 global routine lnk$procsobj (modrfa) =
2948 3412 1
2949 3413 1 READ AND PROCESS ALL REQUIRED OBJECT MODULE RECORDS OF THE FILE JUST OPENED
2950 3414 1 THAT IS, KEEP READING RECORDS TO END OF FILE EXCEPT IF THIS FILE IS A LIBRARY,
2951 3415 1 STOP AT THE FIRST END OF MODULE RECORD.
2952 3416 1
2953 3417 1 MODRFA EXISTS IN THE CASE OF A LIBRARY MODULE OR SHAREABLE IMAGE AND
2954 3418 1 IS A POINTER TO THE RECORD FILE ADDRESS OF THE MODULE.
2955 3419 1
2956 3420 2 begin
2957 3421 2 map
2958 3422 2     modrfa : ref block [, byte] ; ! REALLY A POINTER
2959 3423 2 bind
2960 3424 2     auxfnb = lnk$gl_curfil [fdb$t_auxfnb] : block [, byte]; ! AUXILIARY FILENAME BLOCK
2961 3425 2
2962 3426 2 while lnk$nextrec (objrecdesc) ! IF THERE IS ANOTHER RECORD
2963 3427 3 do begin
2964 3428 3     local status ;
2965 3429 3
2966 3430 3     if .reclng gtru .maxreclng ! AND ITS LENGTH IS ILLEGAL
2967 3431 4 then begin
2968 3432 4         if .mhdseen
2969 3433 4             then signal (lin$ illreclen, 3 ! TELL MODULE NAME IF IT IS KNOWN
2970 3434 4                 ,.reclng, obmodesc [omd$b_namlng]
2971 3435 4                 ,lnk$gl_curfil [fdb$q_filename]
2972 3436 4             )
2973 3437 4         else signal (lin$ reclng, 2 ! OUTPUT AN ERROR AND QUIT ON SEVERE ERROR
2974 3438 4                 ,lnk$gl_curfil [fdb$q_filename]
2975 3439 4                 ,.reclng
2976 3440 4                 ) ;
2977 3441 4     end
2978 3442 4 else begin ! LENGTH OF RECORD OR SO
2979 3443 4     lastrectyp = .currenttyp ; ! COPY OLD CURRENT TO LAST TYPE
2980 3444 4     currenttyp = .objrec [obj$b_rectyp] ; ! AND GET NEW TYPE
2981 3445 4     if .currenttyp lssu .recdispatch [-1] ! CHECK IT IS WITH RANGE
2982 3446 5     and not (not .mhdseen and (.currenttyp neq obj$c_hdr)) ! MUST BE "hdr" IF HEADER NOT SEEN YET
2983 3447 4     then (.recdispatch [.currenttyp]) () ! DISPATCH TO RECORD SPECIFIC ROUTINE
2984 3448 5 else begin
2985 3449 5     if .mhdseen
2986 3450 5     then signal (lin$ illrectyp, 4 ! TELL MODULE NAME IF KNOWN
2987 3451 5         ,.currenttyp, obmodesc [omd$b_namlng]
2988 3452 5         ,lnk$gl_record, lnk$gl_curfil [fdb$q_filename]
2989 3453 5     )
2990 3454 5     else signal (lin$ rectyp, 3 ! ELSE OUTPUT ERROR
2991 3455 5         ,lnk$gl_curfil [fdb$q_filename]
2992 3456 5         ,lnk$gl_record, .currenttyp
2993 3457 5     ) ;
2994 3458 4     end ;
2995 3459 3 end ;
2996 3460 3
2997 3461 4 if ((.currenttyp eql obj$c_eom) or (.currenttyp eql obj$c_eomw)) ! IF THAT RECORD WAS AN EOM
2998 3462 3 and ! OR CONTAINS A SHAREABLE IMAGE
2999 3463 4 (.lnk$gl_curfil [fdb$v_libr] or .lastobjmod [omd$v_shrimg]) ! AND THE FILE IS A LIBRARY
3000 3464 3 then
3001 3465 4 begin ! THEN THE FILE IS DONE
3002 3466 4     lastobjmod [omd$l_modvbn] = .modrfa [rfa$l_vbn] ; ! AS SOON AS WE EXTRACT THE RFA
3003 3467 4     lastobjmod [omd$w_bytoff] = .modrfa [rfa$w_offset] ; ! OF THE MODULE
```

```

3004      3468      4      return true ;
3005      3469      3      end ;
3006      3470      2      end ;
3007      3471      2      ! OF RECORDS LOOP
3008      3472      2      :
3009      3473      2      ALL OBJECT RECORDS HAVE BEEN READ AND PROCESSED
3010      3474      2      :
3011      3475      2      if (.currenttyp neq obj$c_eom)
3012      3476      2      and (.currenttyp neq obj$c_eomw)
3013      3477      2      then begin
3014      3478      2          local      eomrec : block [eom$c_eommin, byte] ;
3015      3479      2
3016      3480      2          signal (lin$noeom, 2
3017      3481      2              ,obmodesc [omd$b_namlng]
3018      3482      2              ,lnk$gl_curfil [fdb$b_filename]
3019      3483      2              ) ;
3020      3484      2          eomrec [eom$b_rectyp] = obj$c_eom ;
3021      3485      2          eomrec [eom$b_comcod] = eom$c_error ;
3022      3486      2          reclng      = 2 ;
3023      3487      2          objrec      = eomrec ;
3024      3488      2          currenttyp = obj$c_eom ;
3025      3489      2          proeom (1) ;
3026      3490      2          mhdseen   = false ;
3027      3491      2          lnmseen   = false ;
3028      3492      2          end ;
3029      3493      2
3030      3494      2      return true ;
3031      3495      1      end ;

```

! IF DID NOT END WITH EOM, THEN WARNING

! CREATE FAKE EOM RECORD

! SET LENGTH OF RECORD

! AND IT'S ADDRESS

! SET RECORD TYPE

! FINISH EOM PROCESSING

! RESET MHD FLAGS

! FINALLY RETURN AFTER NO MORE
! OF PASS 1 OBJ READING ROUTINE

			003C 00000	.ENTRY	LNK\$PROCSOBJ, Save R2,R3,R4,R5	3411
			55 00000000G 00 9E 00002	MOVAB	LIB\$SIGNAL, R5	
			54 00000000G 00 9E 00009	MOVAB	LNK\$GL_CURFIL, R4	
			53 00000000' EF 9E 00010	MOVAB	CURRENTYP, R3	
			5E 0A 04 C2 00017	SUBL2	#4, SP	
			0A A3 9F 0001A 1\$:	PUSHAB	OBJRECDISC	3426
			00 01 FB 0001D	CALLS	#1, LNK\$NXTREC	
			03 50 E8 00024	BLBS	R0, 2\$	
			00BE 31 00027	BRW	13\$	
			51 0A A3 3C 0002A 2\$:	MOVZWL	RECLNG, R1	3430
			51 06 A3 B1 0002E	CMPL	MAXRECLNG, R1	
			28 1E 00032	BGEQU	4\$	
			64 14 C1 00034	ADDL3	#20, LNK\$GL_CURFIL, R0	3435
			11 FD A3 E9 00038	BLBC	MHDSEEN, 3\$	3432
			50 DD 0003C	PUSHL	R0	3435
			66 A3 9F 0003E	PUSHAB	OBMODESC+40	3434
			51 DD 00041	PUSHL	R1	3435
			03 DD 00043	PUSHL	#3	
			00000000G 8F DD 00045	PUSHL	#LINS_ILLRECLN	
			67 11 0004B	BRB	8\$	
			03 BB 0004D 3\$:	PUSHR	#^M<R0,R1>	3438
			02 DD 0004F	PUSHL	#2	
			00000000G 8F DD 00051	PUSHL	#LINS_RECLNG	
			65 04 FB 00057	CALLS	#4, LIB\$SIGNAL	

	FF	A3		5B	11	0005A	BRB	9\$		3430
		63	OE	63	90	0005C	4\$:	MOV B	CURRECTYP, LASTRECTYP	3443
		52		63	90	00060		MOV B	OBJREC, CURRECTYP	3444
	00000000'	EF		52	9A	00064		MOVZBL	CURRECTYP, R2	3445
				15	D1	00067		CMPL	R2, RECDISPATCH-4	
				15	1E	0006E		BGEQU	6\$	
		04	FD	A3	E8	00070		BLBS	MHDSEEN, 5\$	3446
				52	D5	00074		TSTL	R2	
				0D	12	00076		BNEQ	6\$	
		50	00000000'EF	42	D0	00078	5\$:	MOVL	RECDISPATCH[R2], R0	3447
		60		00	FB	00080		CALLS	#0, (R0)	
				32	11	00083		BRB	9\$	
51		64		14	C1	00085	6\$:	ADDL3	#20, LNK\$GL_CURFIL, R1	3452
		50	00000000G	00	D0	00089		MOVL	LNK\$GL_RECORD, R0	
		14	FD	A3	E9	00090		BLBC	MHDSEEN, 7\$	3449
				03	BB	00094		PUSHR	#*M<R0,R1>	3452
			66	A3	9F	00096		PUSHAB	OBMODESC+40	3451
				52	DD	00099		PUSHL	R2	3452
				04	DD	0009B		PUSHL	#4	
			00000000G	8F	DD	0009D		PUSHL	#LINS_ILLRECTYP	
		65		06	FB	000A3		CALLS	#6, LIB\$SIGNAL	
				0F	11	000A6		BRB	9\$	
				05	BB	000A8	7\$:	PUSHR	#*M<R0,R2>	3456
				51	DD	000AA		PUSHL	R1	3455
				03	DD	000AC		PUSHL	#3	
			00000000G	8F	DD	000AE		PUSHL	#LINS_RECTYP	
		65		05	FB	000B4	8\$:	CALLS	#5, LIB\$SIGNAL	
		03		63	91	000B7	9\$:	CMPB	CURRECTYP, #3	3461
				08	13	000BA		BEQL	11\$	
		07		63	91	000BC		CMPB	CURRECTYP, #7	
				03	13	000BF		BEQL	11\$	
			FF	56	31	000C1	10\$:	BRW	1\$	
09	OA	50		64	D0	000C4	11\$:	MOVL	LNK\$GL_CURFIL, R0	3463
		A0		01	E0	000C7		BBS	#1, 10(R0), 12\$	
EC		50	F6	A3	D0	000CC		MOVL	LASTOBJMOD, R0	
		A0		02	E1	000D0		BBC	#2, 20(R0), 10\$	
		51	F6	A3	D0	000D5	12\$:	MOVL	LASTOBJMOD, R1	3466
		50	04	AC	D0	000D9		MOVL	MODRFA, R0	
	OC	A1		60	D0	000DD		MOVL	(R0), 12(R1)	
	10	A1	04	A0	B0	000E1		MOVW	4(R0), 16(R1)	3467
				36	11	000E6		BRB	14\$	3468
		03		63	91	000E8	13\$:	CMPB	CURRECTYP, #3	3475
				31	13	000EB		BEQL	14\$	
		07		63	91	000ED		CMPB	CURRECTYP, #7	3476
				2C	13	000F0		BEQL	14\$	
7E		64		14	C1	000F2		ADDL3	#20, LNK\$GL_CURFIL, -(SP)	3482
			66	A3	9F	000F6		PUSHAB	OBMODESC+40	3481
				02	DD	000F9		PUSHL	#2	3482
			00000000G	8F	DD	000FB		PUSHL	#LINS_NOEOM	
		65		04	FB	00101		CALLS	#4, LIB\$SIGNAL	
		6E	0203	8F	B0	00104		MOVW	#515, EOMREC	3484
	OA	A3		02	B0	00109		MOVW	#2, RECLNG	3486
	OE	A3		6E	9E	0010D		MOVAB	EOMREC, OBJREC	3487
		63		03	90	00111		MOV B	#3, CURRECTYP	3488
				01	DD	00114		PUSHL	#1	3489
	F42A	CF		01	FB	00116		CALLS	#1, PROEOM	
			FD	A3	B4	0011B		CLRW	MHDSEEN	3490

LNK_OBJPASS1
V04=000

I 4
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 115
(34)

50

01 D0 0011E 14\$: MOVL #1, R0
04 00121 RET

: 3494
: 3495

; Routine Size: 290 bytes, Routine Base: \$CODE\$ + 1D29

; 3032 3496 1


```
3034 3497 1 global routine lnk$objpass1 (arglist) =
3035 3498 1
3036 3499 1     THIS IS THE MAIN DRIVING ROUTINE OF PASS ONE. IT OPENS
3037 3500 1     EACH FILE AND IF A LIBRARY, CALLS THE LIBRARY PROCESSING ROUTINE
3038 3501 1     OR IF NOT A LIBRARY ASSUMES IT IS A CONCATENATED OBJ FILE
3039 3502 1     AND CALLS THE OBJ MODULE PROCESSOR.
3040 3503 1     WHEN NO MORE FILES REMAIN, IF THERE ARE STILL UNDEFINED SYMBOLS
3041 3504 1     THE SYSTEM LIBRARY IS MATERIALIZED AS AN INPUT FILE AND
3042 3505 1     PROCESSED.
3043 3506 1
3044 3507 1     ARGLIST CONTAINS THE ADDRESS OF THE ARGUMENT LIST WITH WHICH THE LINKER
3045 3508 1     WAS ORIGINALLY CALLED. IT IS MERELY PASSED ON TO THE OBJECT
3046 3509 1     LIBRARY PROCESSOR SO IT CAN RE-CALL CLI TO OBTAIN THE NAMES (IF ANY)
3047 3510 1     OF MODULES TO BE EXPLICITLY EXTRACTED FROM THE LIBRARY.
3048 3511 1
3049 3512 2 begin
3050 3513 2 builtin
3051 3514 2     insque,                ! QUE INSERTION AND
3052 3515 2     remque;                ! QUE REMOVING INSTRUCTIONS
3053 3516 2 local
3054 3517 2     suffix,                ! UNDEFINED SYMBOL LIST SUFFIX POINTER
3055 3518 2     ntxtsymnam : ref block [, byte], ! NEXT SYMBOL NAME BLOCK POINTER
3056 3519 2     ntxtsymnt : ref block [, byte]; ! FOR SCANNING LIST OF UNDEFINED SYMBOLS
3057 3520 2
3058 3521 2
3059 3522 2     INSERT THE SYMBOL "SYSSK VERSION" INTO THE SYMBOL TABLE AS AN UNDEFINED
3060 3523 2     WEAK ABSOLUTE REFERENCE WITH THE SPECIAL INTERNALLY CREATED SYMBOL
3061 3524 2     FLAG.
3062 3525 2
3063 3526 2 lnk$search (lnk$gt_sysver, symtabent, symntnam); ! FIND WHERE IT GOES
3064 3527 2 lnk$insert (lnk$gt_sysver, symtabent, symntnam); ! AND INSERT IT
3065 3528 2 symtabent [sym$v_weak] = true; ! FLAG WEAK
3066 3529 2 symtabent [sym$v_intsym] = true; ! AND INTERNALLY CREATED
3067 3530 2 lnk$insudfsym (.symtabent); ! INSERT IN UDEFINED SYMBOL LIST
3068 3531 2 obmodesc [omd$l_lsterr] = obmodesc [omd$l_errtxt];
3069 3532 2 lnk$gl_curomd = obmodesc; ! POINT TO THE DESCRIPTOR
3070 3533 2
3071 3534 2
3072 3535 2 BEGIN THE LOOP WHICH OPENS EACH INPUT FILE
3073 3536 2
3074 3537 2 while lnk$nextfil () ! WHILE THERE REMAIN
3075 3538 3 do begin ! MORE OBJ FILES
3076 3539 3     last_lnkopt_fdb = 0; ! Haven't seen linker opt recs in this file
3077 3540 3     lnk$gw_nfiles = lnk$gw_nfiles + 1; ! COUNT THE FILE
3078 3541 3     lastobjmod = lnk$gl_curfil [fdb$l_omdlst]; ! INITIALIZE LAST DESCRIPTOR TO LISTHEAD
3079 3542 3
3080 3543 4     if not (if .lnk$gl_curfil [fdb$v_libr] ! IF THIS INPUT FILE IS A LIBRARY
3081 3544 4         then lnk$procslib (.arglist) ! GO PROCESS IT
3082 3545 4         else if .lnk$gl_curfil [fdb$v_shr] ! IF A SHAREABLE IMAGE
3083 3546 4         then lnk$procshrim () ! GO PROCESS IT
3084 3547 4         else lnk$procsobj (); ! OTHERWISE ASSUME CONCATENATED OBJECT
3085 3548 4     )
3086 3549 3     then return false; ! EXIT NOW ON FAILURE
3087 3550 3
3088 3551 3     lastobjmod [omd$l_nxtomd] = 0; ! END OF USER INPUT FILES
3089 3552 2 end; ! END OF USER INPUT FILES
3090 3553 2
```

```
3091 3554 2 |
3092 3555 2 | NOW CHECK THAT WE DID GET SOME MODULES AND IF SO
3093 3556 2 | CHECK FOR UNDEFINED SYMBOLS.
3094 3557 2 |
3095 3558 2 | lnk$gl_curomd = 0;
3096 3559 2 |
3097 3560 2 | if .lnk$gw_nmodules eql 0 then signal_stop (lin$_nomods);
3098 3561 2 |
3099 3562 2 | if .lnk$gw_nudfsyms neq 0
3100 3563 2 | then signal (lin$_nudfsyms, 1, .lnk$gw_nudfsyms);
3101 3564 2 |
3102 3565 2 | while not remque (.lnk$gl_udflst, ntxsyment)
3103 3566 2 | do begin
3104 3567 3 |   bind ntxsymnam = .ntxsyment - .ntxsyment [sym$b_namlng]
3105 3568 3 |   - snb$c_fxdlen : block [, byte];
3106 3569 3 |
3107 3570 3 |   if not .ntxsyment [sym$v_intsym]
3108 3571 4 |   then begin
3109 3572 4 |     if not .ntxsyment [sym$v_weak]
3110 3573 4 |     then suffix = cstring ('')
3111 3574 5 |     else begin
3112 3575 5 |       suffix = cstring (' (Weak Reference)');
3113 3576 5 |       ntxsyment [sym$v_uf] = true;
3114 3577 4 |     end;
3115 3578 4 |
3116 3579 4 |   if .lnk$gw_nudfsyms neq 0
3117 3580 4 |   then signal (lin$_udfsym, 2
3118 3581 4 |     , ntxsymnam [snb$b_namlng], .suffix
3119 3582 4 |     );
3120 3583 4 |
3121 3584 4 |   ntxsyment [sym$l_value] = 0;
3122 3585 4 |   ntxsyment [sym$v_rel] = false;
3123 3586 4 |
3124 3587 4 |   if not .lnk$gl_ctlmsk [lnk$v_brief]
3125 3588 4 |   and .lnk$gl_ctlmsk [lnk$v_map]
3126 3589 4 |   then crf$insrtkey ( lnk$a1_sytblfmt
3127 3590 4 |     , ntxsymnam [snb$b_namlng]
3128 3591 4 |     , ntxsyment [sym$l_value]
3129 3592 4 |     , ntxsyment [sym$w_flags]
3130 3593 4 |     );
3131 3594 4 |
3132 3595 4 |   else begin
3133 3596 4 |     ntxsyment [sym$l_value] = 0;
3134 3597 4 |     ntxsyment [sym$v_rel] = false;
3135 3598 3 |   end;
3136 3599 2 | end;
3137 3600 2 |
3138 3601 2 |
3139 3602 2 | LIST ANY UNDEFINED ENVIRONMENTS
3140 3603 2 |
3141 3604 2 | if .lnk$gw_nudfenvs neq 0
3142 3605 2 | then begin
3143 3606 3 |   local envdesc : ref block [, byte];
3144 3607 3 |
3145 3608 3 |   signal (lin$_nudfenvs, 1, .lnk$gw_nudfenvs);
3146 3609 3 |
3147 3610 3 |   envdesc = .lnk$gl_udfenv [0];
```

! NO OBJECT MODULE NOW

! IF THERE ARE STILL SOME UNDEFINED (STRONG)
! THEN REPORT THE NUMBER

! AND REMOVE EACH FROM TOP
! OF THE LIST, OUTPUTTING
! POINT TO SYMBOL NAME BLOCK

! THE SYMBOL NAME
! WITH NULL OR 'WEAK'

! SUFFIX DEPENDING ON THE REFERENCE
! BEING WEAK OR STRONG AND
! ENSURE WEAKS ARE NOW DEFINED

! NOW LIST THE SYMBOL PROVIDED
! NOT ALL WERE WEAK

! SET ITS VALUE TO 0
! AND MAKE IT ABSOLUTE

! PROVIDED THIS IS NOT A
! BRIEF MAP
! INSERT THE KEY (SYMBOL) IN
! CREF'S TABLE SO THAT WE MAY
! MAP UNDEFINED SYMBOLS
! AND IGNORE ERROR FROM CREF

! NOT INTERNALLY CREATED SYMBOL

! INTERNALLY CREATED--ZERO VALUE
! AND MAKE IT ABSOLUTE


```
3148 3611 3 while .envdesc neq lnk$gl_udfenv
3149 3612 4 do begin
3150 3613 4 signal (lin$udfenv, 1, envdesc [nvd$b_namlng]);
3151 3614 4 envdesc = .envdesc [nvd$l_udflink];
3152 3615 3 end;
3153 3616 2 end;
3154 3617 2
3155 3618 2
3156 3619 2
3157 3620 2
3158 3621 2 if .lnk$gw_nudflsyms neq 0
3159 3622 2 then begin
3160 3623 3 signal (lin$_nudflsyms, 1, .lnk$gw_nudflsyms);
3161 3624 3
3162 3625 3 while not remque (.lnk$gl_udflsy, ntxsyment)
3163 3626 4 do begin
3164 3627 4 bind ntxsymnam = .ntxsyment - .ntxsyment [sym$b_namlng] - snb$sc_fxdlen : block [, byte];
3165 3628 4
3166 3629 4 if not .ntxsyment [sym$v_weak]
3167 3630 4 then suffix = cstring ('-')
3168 3631 5 else begin
3169 3632 5 suffix = cstring (' (Weak Reference)');
3170 3633 5 ntxsyment [sym$v_def] = true;
3171 3634 4 end;
3172 3635 4
3173 3636 4 signal (lin$_udfsym, 2, ntxsymnam [snb$b_namlng], .suffix);
3174 3637 4 ntxsyment [sym$l_value] = 0;
3175 3638 4 ntxsyment [sym$v_rel] = false;
3176 3639 3 end;
3177 3640 2 end;
3178 3641 2
3179 3642 2 if .lnk$gl_dbgtrfs eq 0
3180 3643 2 then if lnk$search (lnk$gt_imgsta, symtabent, symentnam)
3181 3644 2 then if .symtabent [sym$v_def]
3182 3645 3 then begin
3183 3646 3 lnk$gl_dbgtrf = .symtabent [sym$l_value];
3184 3647 3 lnk$gl_dbgtrfs = 1;
3185 3648 2 end;
3186 3649 2
3187 3650 2
3188 3651 2
3189 3652 2
3190 3653 2
3191 3654 2 lnk$gl_curclu = lnk$gl_defclu;
3192 3655 2 lib$traverse_tree (lnk$gl_pscdflst, checkpddentry);
3193 3656 2
3194 3657 2 return true;
3195 3658 1 end;
```

.PSECT \$PLITS,NOWRT,NOEXE,2

63	6E	65	72	65	66	65	52	20	68	61	65	57	28	01	000A8	P.AAL:	.BYTE	1	
														20	000A9		.ASCII	\ \	
														11	000AA	P.AAM:	.BYTE	17	
														20	000AB		.ASCII	\ (Weak Reference)\	

[illegible]

50		67	3C	000C4	8\$:	MOVZWL	LNK\$GW_NUDFSYMS, R0	3562
		0D	13	000C7		BEQL	9\$	
		50	DD	000C9		PUSHL	R0	3563
		01	DD	000CB		PUSHL	#1	
	00000000G	8F	DD	000CD		PUSHL	#LINS_NUDFSYMS	
69		03	FB	000D3		CALLS	#3, LIB\$SIGNAL	
52	BA	B7	0F	000D6	9\$:	REMQUE	@LNK\$GL_UDFLST, NXTSYMENT	3565
		6B	1D	000DA		BVS	14\$	
50	OF	A2	9A	000DC		MOVZBL	15(NXTSYMENT), R0	3567
52		50	C3	000E0		SUBL3	R0, NXTSYMENT, R0	
53	FB	A0	9E	000E4		MOVAB	-5(R0), R3	3568
54	OA	A2	9E	000E8		MOVAB	10(NXTSYMENT), R4	3570
64		0A	E0	000EC		BBS	#10, (R4), 13\$	
06		64	E8	000F0		BLBS	(R4), 10\$	3572
55	47	A8	9E	000F3		MOVAB	P.AAL, SUFFIX	3573
		07	11	000F7		BRB	11\$	
55	49	A8	9E	000F9	10\$:	MOVAB	P.AAM, SUFFIX	3575
64		02	88	000FD		BISB2	#2, (R4)	3576
		67	B5	00100	11\$:	TSTW	LNK\$GW_NUDFSYMS	3579
		10	13	00102		BEQL	12\$	
		55	DD	00104		PUSHL	SUFFIX	
	04	A3	9F	00106		PUSHAB	4(R3)	3581
		02	DD	00109		PUSHL	#2	
	00000000G	8F	DD	0010B		PUSHL	#LINS_UDFSYM	
69		04	FB	00111		CALLS	#4, LIB\$SIGNAL	
		62	D4	00114	12\$:	CLRL	(NXTSYMENT)	3584
64		08	8A	00116		BICB2	#8, (R4)	3585
B5 00000000G	00	01	E0	00119		BBS	#1, LNK\$GL_CTLMSK+1, 9\$	3587
AD 00000000G	00	04	E1	00121		BBC	#4, LNK\$GL_CTLMSK, 9\$	3588
	7E	64	3C	00129		MOVZWL	(R4), -(SP)	3592
		52	DD	0012C		PUSHL	NXTSYMENT	3591
	04	A3	9F	0012E		PUSHAB	4(R3)	3590
	00000000G	00	9F	00131		PUSHAB	LNK\$AL_SYTBLEMT	3589
00000000G	00	04	FB	00137		CALLS	#4, CRF\$INSRTKEY	3591
		96	11	0013E		BRB	9\$	3570
		62	D4	00140	13\$:	CLRL	(NXTSYMENT)	3596
64		08	8A	00142		BICB2	#8, (R4)	3597
		8F	11	00145		BRB	9\$	3565
50	06	A7	3C	00147	14\$:	MOVZWL	LNK\$GW_NUDFENV, R0	3604
		2D	13	0014B		BEQL	16\$	
		50	DD	0014D		PUSHL	R0	3608
		01	DD	0014F		PUSHL	#1	
	00000000G	8F	DD	00151		PUSHL	#LINS_NUDFENV	
69		03	FB	00157		CALLS	#3, LIB\$SIGNAL	
54	C6	A7	D0	0015A		MOVL	LNK\$GL_UDFENV, ENVDESC	3610
50	C6	A7	9E	0015E	15\$:	MOVAB	LNK\$GL_UDFENV, R0	3611
50		54	D1	00162		CMPL	ENVDESC, R0	
		13	13	00165		BEQL	16\$	
	12	A4	9F	00167		PUSHAB	18(ENVDESC)	3613
		01	DD	0016A		PUSHL	#1	
	00000000G	8F	DD	0016C		PUSHL	#LINS_UDFENV	
69		03	FB	00172		CALLS	#3, LIB\$SIGNAL	
54		64	D0	00175		MOVL	(ENVDESC), ENVDESC	3614
		E4	11	00178		BRB	15\$	3611
50	04	A7	3C	0017A	16\$:	MOVZWL	LNK\$GW_NUDFLSYMS, R0	3621
		48	13	0017E		BEQL	20\$	
		50	DD	00180		PUSHL	R0	3623

			01	DD	00182	PUSHL	#1		
		00000000G	8F	DD	00184	PUSHL	#LINS NUDFLSYMS		
	69		03	FB	0018A	CALLS	#3, LIB\$SIGNAL		
	52	CE	B7	OF	0018D	17\$: REMQUE	@LNK\$GL_UDFLSY, NXTSYMENT		3625
			35	1D	00191	BVS	20\$		
50	50	OF	A2	9A	00193	MOVZBL	15(NXTSYMENT), R0		3627
	52		50	C3	00197	SUBL3	R0, NXTSYMENT, R0		
	50		05	C2	0019B	SUBL2	#5, R0		
	06	0A	A2	E8	0019E	BLBS	10(NXTSYMENT), 18\$		3629
	55	5B	A8	9E	001A2	MOVAB	P.AAN, SUFFIX		3630
			08	11	001A6	BRB	19\$		
	55	5D	A8	9E	001A8	18\$: MOVAB	P.AAO, SUFFIX		3632
0A	A2		02	88	001AC	BISB2	#2, 10(NXTSYMENT)		3633
			55	DD	001B0	19\$: PUSHL	SUFFIX		3636
		04	A0	9F	001B2	PUSHAB	4(R0)		
			02	DD	001B5	PUSHL	#2		
	69	00000000G	8F	DD	001B7	PUSHL	#LINS UDFSVM		
			04	FB	001BD	CALLS	#4, LIB\$SIGNAL		
			62	D4	001C0	CLRL	(NXTSYMENT)		3637
0A	A2		08	8A	001C2	BICB2	#8, 10(NXTSYMENT)		3638
			C5	11	001C6	BRB	17\$		3625
		DE	A7	D5	001C8	20\$: TSTL	LNK\$GL_DBGTFPS		3642
			21	12	001CB	BNEQ	21\$		
		04	A6	9F	001CD	PUSHAB	SYMENTNAM		3643
			56	DD	001D0	PUSHL	R6		
		00000000G	00	9F	001D2	PUSHAB	LNK\$GT_IMGSTA		
	6B		03	FB	001D8	CALLS	#3, LNK\$SEARCH		
	10		50	E9	001DB	BLBC	R0, 21\$		
	50		66	D0	001DE	MOVL	SYMTABENT, R0		3644
08	0A	A0	01	E1	001E1	BBC	#1, 10(R0), 21\$		
	DA	A7	60	D0	001E6	MOVL	(R0), LNK\$GL_DBGTFR		3646
	DE	A7	01	D0	001EA	MOVL	#1, LNK\$GL_DBGTFPS		3647
	00000000G	00	00	9E	001EE	21\$: MOVAB	LNK\$GL_DEFCLU, LNK\$GL_CURCLU		3654
		BC	AA	9F	001F9	PUSHAB	CHECKPDENTRY		3655
		00000000G	00	9F	001FC	PUSHAB	LNK\$GL_PSCDFLST		
	00000000G	00	02	FB	00202	CALLS	#2, LIB\$TRAVERSE_TREE		
		50	01	D0	00209	MOVL	#1, R0		3657
				04	0020C	RET			
			50	D4	0020D	22\$: CLRL	R0		3658
			04	0020F	RET				

; Routine Size: 528 bytes, Routine Base: \$CODE\$ + 1E4B

: 3196	3659	1
: 3197	3660	1
: 3198	3661	1
: 3199	3662	1 end
: 3200	3663	0 eludom

.EXTRN LIB\$SIGNAL, LIB\$STOP

PSECT SUMMARY

LNK_OBJPASS1
V04=000

C 5
16-Sep-1984 00:12:36
14-Sep-1984 12:40:33

VAX-11 Bliss-32 V4.0-742
[LINKER.SRC]LNKOBJPS1.B32;1

Page 122
(35)

Name	Bytes	Attributes
SGLOBALS	86	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
SOWNS	2200	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
SPLITS	208	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
SCODES	8283	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
. ABS .	0	NOVEC, NOWRT, NORD, NOEXE, NOSHR, LCL, ABS, CON, NOPIC, ALIGN(0)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	210	1	1000	00:01.9
\$255\$DUA28:[LINKER.OBJ]DATBAS.L32;1	538	185	34	28	00:00.8

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:LNKOBJPS1/OBJ=OBJ\$:LNKOBJPS1 MSRC\$:LNKOBJPS1/UPDATE=(ENH\$:LNKOBJPS1)

Size: 8283 code + 2494 data bytes
Run Time: 02:34.2
Elapsed Time: 04:59.0
Lines/CPU Min: 1425
Lexemes/CPU-Min: 23122
Memory Used: 540 pages
Compilation Complete

0217

AH-BT13A-SE
 VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

0218 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY